CHAPTER
FOUR

PLANNING OF MANIPULATOR TRAJECTORIES

A mighty maze! but not without a plan.
Alexander Pope

4.1 INTRODUCTION

With the discussion of kinematics and dynamics of a serial link manipulator in the previous chapters as background, we now turn to the problem of controlling the manipulator so that it follows a preplanned path. Before moving a robot arm, it is of considerable interest to know whether there are any obstacles present in its path (obstacle constraint) and whether the manipulator hand must traverse a specified path (path constraint). These two constraints combined give rise to four possible control modes, as tabulated in Table 4.1. From this table, it is noted that the control problem of a manipulator can be conveniently divided into two coherent subproblems—motion (or trajectory) planning and motion control. This chapter focuses attention on various trajectory planning schemes for obstacle-free motion. It also deals with the formalism of describing the desired manipulator motion as sequences of points in space (position and orientation of the manipulator) through which the manipulator must pass, as well as the space curve that it traverses. The space curve that the manipulator hand moves along from the initial location (position and orientation) to the final location is called the *path*. We are interested in developing suitable formalisms for defining and describing the desired motions of the manipulator hand between the path endpoints.

Trajectory planning schemes generally "interpolate" or "approximate" the desired path by a class of polynomial functions and generates a sequence of time-based "control set points" for the control of the manipulator from the initial location to its destination. Path endpoints can be specified either in joint coordinates or in cartesian coordinates. However, they are usually specified in cartesian coordinates because it is easier to visualize the correct end-effector configurations in cartesian coordinates than in joint coordinates. Furthermore, joint coordinates are not suitable as a working coordinate system because the joint axes of most manipulators are not orthogonal and they do not separate position from orientation. If joint coordinates are desired at these locations, then the inverse kinematics solution routine can be called upon to make the necessary conversion.

Quite frequently, there exists a number of possible trajectories between the two given endpoints. For example, one may want to move the manipulator along

Table 4.1 Control modes of a manipulator

		Obstacle constraint	
		Yes	No
Path constraint	Yes	Off-line collision-free path planning plus on-line path tracking	Off-line path planning plus on-line path tracking
	No	Positional control plus on-line obstacle detection and avoidance	Positional control

a straight-line path that connects the endpoints (straight-line trajectory); or to move the manipulator along a smooth, polynomial trajectory that satisfies the position and orientation constraints at both endpoints (joint-interpolated trajectory). In this chapter, we discuss the formalisms for planning both joint-interpolated and straight-line path trajectories. We shall first discuss simple trajectory planning that satisfies path constraints and then extend the concept to include manipulator dynamics constraints.

A systematic approach to the trajectory planning problem is to view the trajectory planner as a black box, as shown in Fig. 4.1. The trajectory planner accepts input variables which indicate the constraints of the path and outputs a sequence of time-based intermediate configurations of the manipulator hand (position and orientation, velocity, and acceleration), expressed either in joint or cartesian coordinates, from the initial location to the final location. Two common approaches are used to plan manipulator trajectories. The first approach requires the user to explicitly specify a set of constraints (e.g., continuity and smoothness) on position, velocity, and acceleration of the manipulator's generalized coordinates at selected locations (called *knot points* or *interpolation points*) along the trajectory. The trajectory planner then selects a parameterized trajectory from a class of functions (usually the class of polynomial functions of degree n or less, for some n, in the time interval $[t_0, t_f]$) that "interpolates" and satisfies the constraints at the interpolation points. In the second approach, the user explicitly specifies the path that the manipulator must traverse by an analytical function, such as a straight-line path in cartesian coordinates, and the trajectory planner determines a desired trajectory either in joint coordinates or cartesian coordinates that approximates the desired path. In the first approach, the constraint specification and the planning of the manipulator trajectory are performed in joint coordinates. Since no constraints are imposed on the manipulator hand, it is difficult for the user to trace the path that

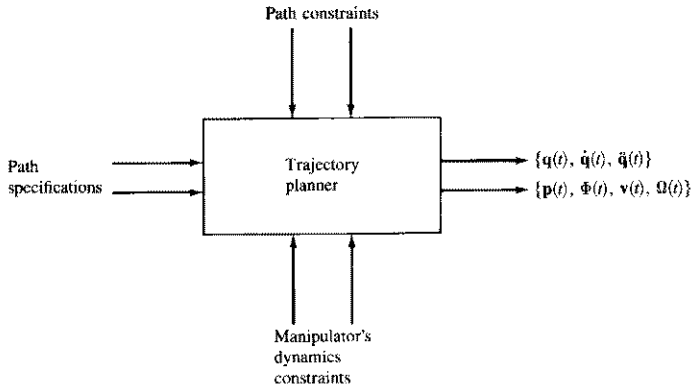

Figure 4.1 Trajectory planner block diagram.

the manipulator hand traverses. Hence, the manipulator hand may hit obstacles with no prior warning. In the second approach, the path constraints are specified in cartesian coordinates, and the joint actuators are servoed in joint coordinates. Hence, to find a trajectory that approximates the desired path closely, one must convert the cartesian path constraints to joint path constraints by some functional approximations and then find a parameterized trajectory that satisfies the joint path constraints.

The above two approaches for planning manipulator trajectories should result in simple trajectories that are meant to be efficient, smooth, and accurate with a fast computation time (near real time) for generating the sequence of control set points along the desired trajectory of the manipulator. However, the sequences of the time-based joint-variable space vectors $\{\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t)\}$ are generated without taking the dynamics of the manipulator into consideration. Thus, large tracking errors may result in the servo control of the manipulator. We shall discuss this problem in Sec. 4.4.3. This chapter begins with a discussion of general issues that arise in trajectory planning in Sec. 4.2; joint-interpolated trajectory in Sec. 4.3; straight-line trajectory planning in Sec. 4.4; and a cubic polynomial trajectory along a straight-line path in joint coordinates with manipulator dynamics taken into consideration in Sec. 4.4.3. Section 4.5 summarizes the results.

4.2 GENERAL CONSIDERATIONS ON TRAJECTORY PLANNING

Trajectory planning can be conducted either in the joint-variable space or in the cartesian space. For joint-variable space planning, the time history of all joint variables and their first two time derivatives are planned to describe the desired motion of the manipulator. For cartesian space planning, the time history of the

manipulator hand's position, velocity, and acceleration are planned, and the corresponding joint positions, velocities, and accelerations are derived from the hand information. Planning in the joint-variable space has three advantages: (1) the trajectory is planned directly in terms of the controlled variables during motion, (2) the trajectory planning can be done in near real time, and (3) the joint trajectories are easier to plan. The associated disadvantage is the difficulty in determining the locations of the various links and the hand during motion, a task that is usually required to guarantee obstacle avoidance along the trajectory.

In general, the basic algorithm for generating joint trajectory set points is quite simple:

$t = t_0$;
loop: Wait for next control interval;
$t = t + \Delta t$;
$\mathbf{h}(t)$ = where the manipulator joint position should be at time t;
If $t = t_f$, then exit;
go to *loop*;

where Δt is the control sampling period for the manipulator.

From the above algorithm, we see that the computation consists of a trajectory function (or trajectory planner) $\mathbf{h}(t)$ which must be updated in every control interval. Thus, four constraints are imposed on the planned trajectory. First, the trajectory set points must be readily calculable in a noniterative manner. Second, intermediate positions must be determined and specified deterministically. Third, the continuity of the joint position and its first two time derivatives must be guaranteed so that the planned joint trajectory is smooth. Finally, extraneous motions, such as "wandering," must be minimized.

The above four constraints on the planned trajectory will be satisfied if the time histories of the joint variables can be specified by polynomial sequences. If the joint trajectory for a given joint (say joint i) uses p polynomials, then $3(p + 1)$ coefficients are required to specify initial and terminal conditions (joint position, velocity, and acceleration) and guarantee continuity of these variables at the polynomial boundaries. If an additional intermediate condition such as position is specified, then an additional coefficient is required for each intermediate condition. In general, two intermediate positions may be specified: one near the initial position for departure and the other near the final position for arrival which will guarantee safe departure and approach directions, in addition to a better controlled motion. Thus, one seventh-degree polynomial for each joint variable connecting the initial and final positions would suffice, as would two quartic and one cubic (4-3-4) trajectory segments, two cubics and one quintic (3-5-3) trajectory segments, or five cubic (3-3-3-3-3) trajectory segments. This will be discussed further in the next section.

For cartesian path control, the above algorithm can be modified to:

$t = t_0$;
loop: Wait for next control interval;
$t = t + \Delta t$;
$\mathbf{H}(t)$ = where the manipulator hand should be at time t;
$\mathbf{Q}[\mathbf{H}(t)]$ = joint solution corresponding to $\mathbf{H}(t)$;
If $t = t_f$, then exit;
go to *loop*;

Here, in addition to the computation of the manipulator hand trajectory function $\mathbf{H}(t)$ at every control interval, we need to convert the cartesian positions into their corresponding joint solutions, $\mathbf{Q}[\mathbf{H}(t)]$. The matrix function $\mathbf{H}(t)$ indicates the desired location of the manipulator hand at time t and can be easily realized by a 4×4 transformation matrix, as discussed in Sec. 4.4.

Generally, cartesian path planning can be realized in two coherent steps: (1) generating or selecting a set of knot points or interpolation points in cartesian coordinates according to some rules along the cartesian path and then (2) specifying a class of functions to link these knot points (or to approximate these path segments) according to some criteria. For the latter step, the criteria chosen are quite often dictated by the following control algorithms to ensure the desired path tracking. There are two major approaches for achieving it: (1) The cartesian space-oriented method in which most of the computation and optimization is performed in cartesian coordinates and the subsequent control is performed at the hand level.† The servo sample points on the desired straight-line path are selected at a fixed servo interval and are converted into their corresponding joint solutions in real time while controlling the manipulator. The resultant trajectory is a piecewise straight line. Paul [1979], Taylor [1979], and Luh and Lin [1981] all reported methods for using a straight line to link adjacent cartesian knot points. (2) The joint space-oriented method in which a low-degree polynomial function in the joint-variable space is used to approximate the path segment bounded by two adjacent knot points on the straight-line path and the resultant control is done at the joint level.‡ The resultant cartesian path is a nonpiecewise straight line. Taylor's bounded deviation joint path (Taylor [1979]) and Lin's cubic polynomial trajectory method (Lin et al. [1983]) all used low-degree polynomials in the joint-variable space to approximate the straight-line path.

The cartesian space-oriented method has the advantage of being a straightforward concept, and a certain degree of accuracy is assured along the desired straight-line path. However, since all the available control algorithms are invariably based on joint coordinates because, at this time, there are no sensors capable

† The error actuating signal to the joint actuators is computed based on the error between the target cartesian position and the actual cartesian position of the manipulator hand.

‡ The error actuating signal to the joint actuators is computed based on the error between the target joint position and the actual joint position of the manipulator hand.

of measuring the manipulator hand in cartesian coordinates, cartesian space path planning requires transformations between the cartesian and joint coordinates in real time—a task that is computationally intensive and quite often leads to longer control intervals. Furthermore, the transformation from cartesian coordinates to joint coordinates is ill-defined because it is not a one-to-one mapping. In addition, if manipulator dynamics are included in the trajectory planning stage, then path constraints are specified in cartesian coordinates while physical constraints, such as torque and force, velocity, and acceleration limits of each joint motor, are bounded in joint coordinates. Thus, the resulting optimization problem will have mixed constraints in two different coordinate systems.

Because of the various disadvantages mentioned above, the joint space-oriented method, which converts the cartesian knot points into their corresponding joint coordinates and uses low-degree polynomials to interpolate these joint knot points, is widely used. This approach has the advantages of being computationally faster and makes it easier to deal with the manipulator dynamics constraints. However, it loses accuracy along the cartesian path when the sampling points fall on the fitted, smooth polynomials. We shall examine several planning schemes in these approaches in Sec. 4.4.

4.3 JOINT-INTERPOLATED TRAJECTORIES

To servo a manipulator, it is required that its robot arm's configuration at both the initial and final locations must be specified before the motion trajectory is planned. In planning a joint-interpolated motion trajectory for a robot arm, Paul [1972] showed that the following considerations are of interest:

1. When picking up an object, the motion of the hand must be directed away from an object; otherwise the hand may crash into the supporting surface of the object.
2. If we specify a departure position (lift-off point) along the normal vector to the surface out from the initial position and if we require the hand (i.e., the origin of the hand coordinate frame) to pass through this position, we then have an admissible departure motion. If we further specify the time required to reach this position, we could then control the speed at which the object is to be lifted.
3. The same set of lift-off requirements for the arm motion is also true for the set-down point of the final position motion (i.e., we must move to a normal point out from the surface and then slow down to the final position) so that the correct approach direction can be obtained and controlled.
4. From the above, we have four positions for each arm motion: initial, lift-off, set-down, and final (see Fig. 4.2).
5. Position constraints
   (*a*) Initial position: velocity and acceleration are given (normally zero).
   (*b*) Lift-off position: continuous motion for intermediate points.
   (*c*) Set-down position: same as lift-off position.
   (*d*) Final position: velocity and acceleration are given (normally zero).

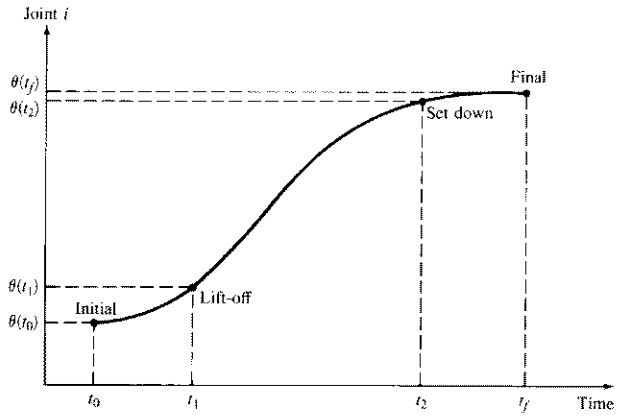

Figure 4.2 Position conditions for a joint trajectory.

6. In addition to these constraints, the extrema of all the joint trajectories must be within the physical and geometric limits of each joint.
7. Time considerations
   (*a*) Initial and final trajectory segments: time is based on the rate of approach of the hand to and from the surface and is some fixed constant based on the characteristics of the joint motors.
   (*b*) Intermediate points or midtrajectory segment: time is based on maximum velocity and acceleration of the joints, and the maximum of these times is used (i.e., the maximum time of the slowest joint is used for normalization).

The constraints of a typical joint trajectory are listed in Table 4.2. Based on these constraints, we are concerned with selecting a class of polynomial functions of degree n or less such that the required joint position, velocity, and acceleration at these knot points (initial, lift-off, set-down, and final positions) are satisfied, and the joint position, velocity, and acceleration are continuous on the entire time interval $[t_0, t_f]$. One approach is to specify a seventh-degree polynomial for each joint i,

$$q_i(t) = a_7 t^7 + a_6 t^6 + a_5 t^5 + a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 \qquad (4.3\text{-}1)$$

where the unknown coefficients a_j can be determined from the known positions and continuity conditions. However, the use of such a high-degree polynomial to interpolate the given knot points may not be satisfactory. It is difficult to find its extrema and it tends to have extraneous motion. An alternative approach is to split the entire joint trajectory into several trajectory segments so that different interpolating polynomials of a lower degree can be used to interpolate in each trajectory

Table 4.2 Constraints for planning joint-interpolated trajectory

Initial position:

1. Position (given)
2. Velocity (given, normally zero)
3. Acceleration (given, normally zero)

Intermediate positions:

4. Lift-off position (given)
5. Lift-off position (continuous with previous trajectory segment)
6. Velocity (continuous with previous trajectory segment)
7. Acceleration (continuous with previous trajectory segment)
8. Set-down position (given)
9. Set-down position (continuous with next trajectory segment)
10. Velocity (continuous with next trajectory segment)
11. Acceleration (continuous with next trajectory segment)

Final position:

12. Position (given)
13. Velocity (given, normally zero)
14. Acceleration (given, normally zero)

segment. There are different ways a joint trajectory can be split, and each method possesses different properties. The most common methods are the following:

4-3-4 ***Trajectory***. Each joint has the following three trajectory segments: the first segment is a fourth-degree polynomial specifying the trajectory from the initial position to the lift-off position. The second trajectory segment (or midtrajectory segment) is a third-degree polynomial specifying the trajectory from the lift-off position to the set-down position. The last trajectory segment is a fourth-degree polynomial specifying the trajectory from the set-down position to the final position.

3-5-3 ***Trajectory***. Same as 4-3-4 trajectory, but uses polynomials of different degrees for each segment: a third-degree polynomial for the first segment, a fifth-degree polynomial for the second segment, and a third-degree polynomial for the last segment.

5-*Cubic Trajectory*. Cubic spline functions of third-degree polynomials for five trajectory segments are used.

Note that the foregoing discussion is valid for each joint trajectory; that is, each joint trajectory is split into either a three-segment or a five-segment trajectory. The number of polynomials for a 4-3-4 trajectory of an N-joint manipulator will have N joint trajectories or $N \times 3 = 3N$ trajectory segments and $7N$ polynomial coefficients to evaluate plus the extrema of the $3N$ trajectory segments. We

shall discuss the planning of a 4-3-4 joint trajectory and a 5-cubic joint trajectory in the next section.

4.3.1 Calculation of a 4-3-4 Joint Trajectory

Since we are determining N joint trajectories in each trajectory segment, it is convenient to introduce a normalized time variable, $t \in [0, 1]$, which allows us to treat the equations of each trajectory segment for each joint angle in the same way, with time varying from $t = 0$ (initial time for all trajectory segments) to $t = 1$ (final time for all trajectory segments). Let us define the following variables:

t : normalized time variable, $t \in [0, 1]$
τ : real time in seconds
τ_i : real time at the end of the ith trajectory segment
$t_i = \tau_i - \tau_{i-1}$: real time required to travel through the ith segment

$$t := \frac{\tau - \tau_{i-1}}{\tau_i - \tau_{i-1}}; \quad \tau \in [\tau_{i-1}, \tau_i]; \quad t \in [0, 1]$$

The trajectory consists of the polynomial sequences, $h_i(t)$, which together form the trajectory for joint j. The polynomial equations for each joint variable in each trajectory segment expressed in normalized time are:

$$h_1(t) = a_{14}t^4 + a_{13}t^3 + a_{12}t^2 + a_{11}t + a_{10} \qquad \text{(1st segment)} \qquad (4.3\text{-}2)$$

$$h_2(t) = a_{23}t^3 + a_{22}t^2 + a_{21}t + a_{20} \qquad \text{(2nd segment)} \qquad (4.3\text{-}3)$$

and

$$h_n(t) = a_{n4}t^4 + a_{n3}t^3 + a_{n2}t^2 + a_{n1}t + a_{n0} \qquad \text{(last segment)} \qquad (4.3\text{-}4)$$

The subscript of each polynomial equation indicates the segment number, and n indicates the last trajectory segment. The unknown coefficient a_{ji} indicates the ith coefficient for the j trajectory segment of a joint trajectory. The boundary conditions that this set of joint trajectory segment polynomials must satisfy are:

1. Initial position $= \theta_0 = \theta(t_0)$
2. Magnitude of initial velocity $= v_0$ (normally zero)
3. Magnitude of initial acceleration $= a_0$ (normally zero)
4. Lift-off position $= \theta_1 = \theta(t_1)$
5. Continuity in position at t_1 [that is, $\theta(t_1^-) = \theta(t_1^+)$]
6. Continuity in velocity at t_1 [that is, $v(t_1^-) = v(t_1^+)$]
7. Continuity in acceleration at t_1 [that is, $a(t_1^-) = a(t_1^+)$]
8. Set-down position $= \theta_2 = \theta(t_2)$
9. Continuity in position at t_2 [that is, $\theta(t_2^-) = \theta(t_2^+)$]
10. Continuity in velocity at t_2 [that is, $v(t_2^-) = v(t_2^+)$]
11. Continuity in acceleration at t_2 [that is, $a(t_2^-) = a(t_2^+)$]
12. Final position $= \theta_f = \theta(t_f)$

13. Magnitude of final velocity $= v_f$ (normally zero)
14. Magnitude of final acceleration $= a_f$ (normally zero)

The boundary conditions for the 4-3-4 joint trajectory are shown in Fig. 4.3. The first and second derivatives of these polynomial equations with respect to real time τ can be written as:

$$v_i(t) = \frac{dh_i(t)}{d\tau} = \frac{dh_i(t)}{dt}\frac{dt}{d\tau} = \frac{1}{\tau_i - \tau_{i-1}}\frac{dh_i(t)}{dt}$$

$$= \frac{1}{t_i}\frac{dh_i(t)}{dt} = \frac{1}{t_i}\dot{h}_i(t) \qquad i = 1, 2, n \tag{4.3-5}$$

and

$$a_i(t) = \frac{d^2h_i(t)}{d\tau^2} = \frac{1}{(\tau_i - \tau_{i-1})^2}\frac{d^2h_i(t)}{dt^2}$$

$$= \frac{1}{t_i^2}\frac{d^2h_i(t)}{dt^2} = \frac{1}{t_i^2}\ddot{h}_i(t) \qquad i = 1, 2, n \tag{4.3-6}$$

For the first trajectory segment, the governing polynomial equation is of the fourth degree:

$$h_1(t) = a_{14}t^4 + a_{13}t^3 + a_{12}t^2 + a_{11}t + a_{10} \qquad t \in [0, 1] \tag{4.3-7}$$

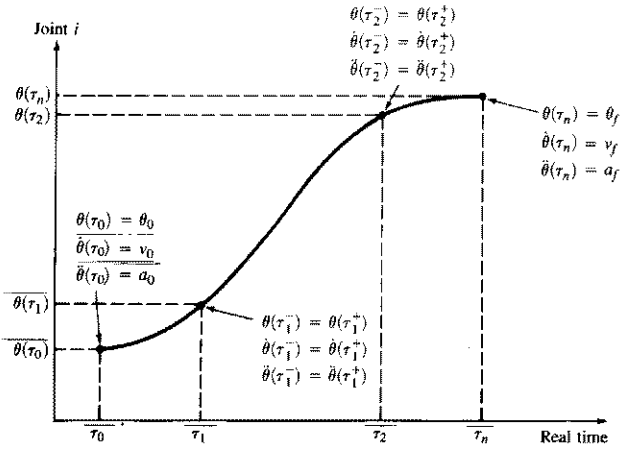

Figure 4.3 Boundary conditions for a 4-3-4 joint trajectory.

From Eqs. (4.3-5) and (4.3-6), its first two time derivatives with respect to real time are

$$v_1(t) = \frac{\dot{h}_1(t)}{t_1} = \frac{4a_{14}t^3 + 3a_{13}t^2 + 2a_{12}t + a_{11}}{t_1} \tag{4.3-8}$$

and

$$a_1(t) = \frac{\ddot{h}_1(t)}{t_1^2} = \frac{12a_{14}t^2 + 6a_{13}t + 2a_{12}}{t_1^2} \tag{4.3-9}$$

1. For $t = 0$ (at the initial position of this trajectory segment). Satisfying the boundary conditions at this position leads to

$$a_{10} = h_1(0) = \theta_0 \qquad \text{(given)} \tag{4.3-10}$$

$$v_0 = \frac{\dot{h}_1(0)}{t_1} = \left[\frac{4a_{14}t^3 + 3a_{13}t^2 + 2a_{12}t + a_{11}}{t_1} \right]_{t=0} = \frac{a_{11}}{t_1} \tag{4.3-11}$$

which gives

$$a_{11} = v_0 t_1$$

and

$$a_0 = \frac{\ddot{h}_1(0)}{t_1^2} = \left[\frac{12a_{14}t^2 + 6a_{13}t + 2a_{12}}{t_1^2} \right]_{t=0} = \frac{2a_{12}}{t_1^2} \tag{4.3-12}$$

which yields

$$a_{12} = \frac{a_0 t_1^2}{2}$$

With these unknowns determined, Eq. (4.3-7) can be rewritten as:

$$h_1(t) = a_{14}t^4 + a_{13}t^3 + \left(\frac{a_0 t_1^2}{2} \right) t^2 + (v_0 t_1)t + \theta_0 \qquad t \in [0, 1] \tag{4.3-13}$$

2. For $t = 1$ (at the final position of this trajectory segment). At this position, we relax the requirement that the interpolating polynomial must pass through the position exactly. We only require that the velocity and acceleration at this position

have to be continuous with the velocity and acceleration, respectively, at the beginning of the next trajectory segment. The velocity and acceleration at this position are:

$$v_1(1) \triangleq v_1 = \frac{\dot{h}_1(1)}{t_1} = \frac{4a_{14} + 3a_{13} + a_0 t_1^2 + v_0 t_1}{t_1} \tag{4.3-14}$$

$$a_1(1) \triangleq a_1 = \frac{\ddot{h}_1(1)}{t_1^2} = \frac{12a_{14} + 6a_{13} + a_0 t_1^2}{t_1^2} \tag{4.3-15}$$

For the second trajectory segment, the governing polynomial equation is of the third degree:

$$h_2(t) = a_{23}t^3 + a_{22}t^2 + a_{21}t + a_{20} \qquad t \in [0, 1] \tag{4.3-16}$$

1. For $t = 0$ (at the lift-off position). Using Eqs. (4.3-5) and (4.3-6), the velocity and acceleration at this position are, respectively,

$$h_2(0) = a_{20} = \theta_2(0) \tag{4.3-17}$$

$$v_1 = \frac{\dot{h}_2(0)}{t_2} = \left[\frac{3a_{23}t^2 + 2a_{22}t + a_{21}}{t_2} \right]_{t=0} = \frac{a_{21}}{t_2} \tag{4.3-18}$$

which gives

$$a_{21} = v_1 t_2$$

and

$$a_1 = \frac{\ddot{h}_2(0)}{t_2^2} = \left[\frac{6a_{23}t + 2a_{22}}{t_2^2} \right]_{t=0} = \frac{2a_{22}}{t_2^2} \tag{4.3-19}$$

which yields

$$a_{22} = \frac{a_1 t_2^2}{2}$$

Since the velocity and acceleration at this position must be continuous with the velocity and acceleration at the end of the previous trajectory segment respectively, we have

$$\frac{\dot{h}_2(0)}{t_2} = \frac{\dot{h}_1(1)}{t_1} \quad \text{and} \quad \frac{\ddot{h}_2(0)}{t_2^2} = \frac{\ddot{h}_1(1)}{t_1^2} \tag{4.3-20}$$

which, respectively, leads to

$$\left(\frac{3a_{23}t^2 + 2a_{22}t + a_{21}}{t_2} \right)_{t=0} = \left(\frac{4a_{14}t^3 + 3a_{13}t^2 + 2a_{12}t + a_{11}}{t_1} \right)_{t=1} \tag{4.3-21}$$

or

$$\frac{-a_{21}}{t_2} + \frac{4a_{14}}{t_1} + \frac{3a_{13}}{t_1} + \frac{a_0 t_1^2}{t_1} + \frac{v_0 t_1}{t_1} = 0 \tag{4.3-22}$$

and

$$\left(\frac{6a_{23}t + 2a_{22}}{t_2^2} \right)_{t=0} = \left(\frac{12a_{14}t^2 + 6a_{13}t + 2a_{12}}{t_1^2} \right)_{t=1} \tag{4.3-23}$$

or

$$\frac{-2a_{22}}{t_2^2} + \frac{12a_{14}}{t_1^2} + \frac{6a_{13}}{t_1^2} + \frac{a_0 t_1^2}{t_1^2} = 0 \tag{4.3-24}$$

2. For $t = 1$ (at the set-down position). Again the velocity and acceleration at this position must be continuous with the velocity and acceleration at the beginning of the next trajectory segment. The velocity and acceleration at this position are obtained, respectively, as:

$$h_2(1) = a_{23} + a_{22} + a_{21} + a_{20} \tag{4.3-25}$$

$$v_2(1) = \frac{\dot{h}_2(1)}{t_2} = \left(\frac{3a_{23}t^2 + 2a_{22}t + a_{21}}{t_2} \right)_{t=1} \tag{4.3-26}$$

$$= \frac{3a_{23} + 2a_{22} + a_{21}}{t_2}$$

and

$$a_2(1) = \frac{\ddot{h}_2(1)}{t_2^2} = \left(\frac{6a_{23}t + 2a_{22}}{t_2^2} \right)_{t=1} = \frac{6a_{23} + 2a_{22}}{t_2^2} \tag{4.3-27}$$

For the last trajectory segment, the governing polynomial equation is of the fourth degree:

$$h_n(t) = a_{n4}t^4 + a_{n3}t^3 + a_{n2}t^2 + a_{n1}t + a_{n0} \qquad t \in [0, 1] \tag{4.3-28}$$

If we substitute $\bar{t} = t - 1$ into t in the above equation, we have shifted the normalized time t from $t \in [0, 1]$ to $\bar{t} \in [-1, 0]$. Then Eq. (4.3-28) becomes

$$h_n(\bar{t}) = a_{n4}\bar{t}^4 + a_{n3}\bar{t}^3 + a_{n2}\bar{t}^2 + a_{n1}\bar{t} + a_{n0} \qquad \bar{t} \in [-1, 0] \tag{4.3-29}$$

Using Eqs. (4.3-5) and (4.3-6), its first and second derivatives with respect to real time are

$$v_n(\bar{t}) = \frac{\dot{h}_n(\bar{t})}{t_n} = \frac{4a_{n4}\bar{t}^3 + 3a_{n3}\bar{t}^2 + 2a_{n2}\bar{t} + a_{n1}}{t_n} \tag{4.3-30}$$

and

$$a_n(\bar{t}) = \frac{\ddot{h}_n(\bar{t})}{t_n^2} = \frac{12a_{n4}\bar{t}^2 + 6a_{n3}\bar{t} + 2a_{n2}}{t_n^2} \tag{4.3-31}$$

1. For $\bar{t} = 0$ (at the final position of this segment). Satisfying the boundary conditions at this final position of the trajectory, we have

$$h_n(0) = a_{n0} = \theta_f \tag{4.3-32}$$

$$v_f = \frac{\dot{h}_n(0)}{t_n} = \frac{a_{n1}}{t_n} \tag{4.3-33}$$

which gives

$$a_{n1} = v_f t_n$$

and

$$a_f = \frac{\ddot{h}_n(0)}{t_n^2} = \frac{2a_{n2}}{t_n^2} \tag{4.3-34}$$

which yields

$$a_{n2} = \frac{a_f t_n^2}{2}$$

2. For $\bar{t} = -1$ (at the starting position of this trajectory segment). Satisfying the boundary conditions at this position, we have, at the set-down position,

$$h_n(-1) = a_{n4} - a_{n3} + \frac{a_f t_n^2}{2} - v_f t_n + \theta_f = \theta_2(1) \tag{4.3-35}$$

and

$$\frac{\dot{h}_n(-1)}{t_n} = \left(\frac{4a_{n4}\bar{t}^3 + 3a_{n3}\bar{t}^2 + 2a_{n2}\bar{t} + a_{n1}}{t_n} \right)_{\bar{t}=-1}$$

$$= \frac{-4a_{n4} + 3a_{n3} - a_f t_n^2 + v_f t_n}{t_n} \tag{4.3-36}$$

and

$$\frac{\ddot{h}_n(-1)}{t_n^2} = \left(\frac{12a_{n4}\bar{t}^2 + 6a_{n3}\bar{t} + 2a_{n2}}{t_n^2} \right)_{\bar{t}=-1}$$

$$= \frac{12a_{n4} - 6a_{n3} + a_f t_n^2}{t_n^2} \tag{4.3-37}$$

The velocity and acceleration continuity conditions at this set-down point are

$$\frac{\dot{h}_2(1)}{t_2} = \frac{\dot{h}_n(-1)}{t_n} \quad \text{and} \quad \frac{\ddot{h}_2(1)}{t_2^2} = \frac{\ddot{h}_n(-1)}{t_n^2} \tag{4.3-38}$$

or

$$\frac{4a_{n4} - 3a_{n3} + a_f t_n^2 - v_f t_n}{t_n} + \frac{3a_{23}}{t_2} + \frac{2a_{22}}{t_2} + \frac{a_{21}}{t_2} = 0 \tag{4.3-39}$$

and

$$\frac{-12a_{n4} + 6a_{n3} - a_f t_n^2}{t_n^2} + \frac{6a_{23}}{t_2^2} + \frac{2a_{22}}{t_2^2} = 0 \tag{4.3-40}$$

The difference of joint angles between successive trajectory segments can be found to be

$$\delta_1 = \theta_1 - \theta_0 = h_1(1) - h_1(0) = a_{14} + a_{13} + \frac{a_0 t_1^2}{2} + v_0 t_1 \tag{4.3-41}$$

$$\delta_2 = \theta_2 - \theta_1 = h_2(1) - h_2(0) = a_{23} + a_{22} + a_{21} \tag{4.3-42}$$

and

$$\delta_n = \theta_f - \theta_2 = h_n(0) - h_n(-1) = -a_{n4} + a_{n3} - \frac{a_f t_n^2}{2} + v_f t_n \tag{4.3-43}$$

All the unknown coefficients of the trajectory polynomial equations can be determined by simultaneously solving Eqs. (4.3-41), (4.3-22), (4.3-24), (4.3-42),

(4.3-39), (4.3-40), and (4.3-43). Rewriting them in matrix vector notation, we have

$$\mathbf{y} = \mathbf{C}\mathbf{x} \tag{4.3-44}$$

where

$$\mathbf{y} = \left[\delta_1 - \frac{a_0 t_1^2}{2} - v_0 t_1 \,,\; -a_0 t_1 - v_0 \,,\; -a_0 \,,\; \delta_2 \,, \right. \tag{4.3-45}$$
$$\left. -a_f t_n + v_f \,,\; a_f \,,\; \delta_n + \frac{a_f t_n^2}{2} - v_f t_n \right]^T$$

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 3/t_1 & 4/t_1 & -1/t_2 & 0 & 0 & 0 & 0 \\ 6/t_1^2 & 12/t_1^2 & 0 & -2/t_2^2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1/t_2 & 2/t_2 & 3/t_2 & -3/t_n & 4/t_n \\ 0 & 0 & 0 & 2/t_2^2 & 6/t_2^2 & 6/t_n^2 & -12/t_n^2 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \tag{4.3-46}$$

and $$\mathbf{x} = (a_{13} \,,\; a_{14} \,,\; a_{21} \,,\; a_{22} \,,\; a_{23} \,,\; a_{n3} \,,\; a_{n4})^T \tag{4.3-47}$$

Then the planning of the joint trajectory (for each joint) reduces to solving the matrix vector equation in Eq. (4.3-44):

$$y_i = \sum_{j=1}^{7} c_{ij} x_j \tag{4.3-48}$$

or $$\mathbf{x} = \mathbf{C}^{-1}\mathbf{y} \tag{4.3-49}$$

The structure of matrix $\mathbf{C}$ makes it easy to compute the unknown coefficients and the inverse of $\mathbf{C}$ always exists if the time intervals t_i, $i = 1, 2, n$ are positive values. Solving Eq. (4.3-49), we obtain all the coefficients for the polynomial equations for the joint trajectory segments for joint j.

Since we made a change in normalized time to run from $[0, 1]$ to $[-1, 0]$ for the last trajectory segment, after obtaining the coefficients a_{ni} from the above

matrix equation, we need to reconvert the normalized time back to [0, 1]. This can be accomplished by substituting $t = \bar{t} + 1$ into $\bar{t}$ in Eq. (4.3-29). Thus we obtain

$$\begin{aligned} h_n(t) &= a_{n4}t^4 + (-4a_{n4} + a_{n3})t^3 + (6a_{n4} - 3a_{n3} + a_{n2})t^2 \\ &\quad + (-4a_{n4} + 3a_{n3} - 2a_{n2} + a_{n1})t \\ &\quad + (a_{n4} - a_{n3} + a_{n2} - a_{n1} + a_{n0}) \qquad t \in [0, 1] \end{aligned} \tag{4.3-50}$$

The resulting polynomial equations for the 4-3-4 trajectory, obtained by solving the above matrix equation, are listed in Table 4.3. Similarly, we can apply this technique to compute a 3-5-3 joint trajectory. This is left as an exercise to the reader. The polynomial equations for a 3-5-3 joint trajectory are listed in Table 4.4.

4.3.2 Cubic Spline Trajectory (Five Cubics)

The interpolation of a given function by a set of cubic polynomials, preserving continuity in the first and second derivatives at the interpolation points is known as cubic spline functions. The degree of approximation and smoothness that can be achieved is relatively good. In general, a spline curve is a polynomial of degree k with continuity of derivative of order $k - 1$, at the interpolation points. In the case of cubic splines, the first derivative represents continuity in the velocity and the second derivative represents continuity in the acceleration. Cubic splines offer several advantages. First, it is the lowest degree polynomial function that allows continuity in velocity and acceleration. Second, low-degree polynomials reduce the effort of computations and the possibility of numerical instabilities.

The general equation of five-cubic polynomials for each joint trajectory segment is:

$$h_j(t) = a_{j3}t^3 + a_{j2}t^2 + a_{j1}t + a_{j0} \qquad j = 1, 2, 3, 4, n \tag{4.3-51}$$

with $\tau_{j-1} \leqslant \tau \leqslant \tau_j$ and $t \in [0, 1]$. The unknown coefficient a_{ji} indicates the ith coefficient for joint j trajectory segment and n indicates the last trajectory segment.

In using five-cubic polynomial interpolation, we need to have five trajectory segments and six interpolation points. However, from our previous discussion, we only have four positions for interpolation, namely, initial, lift-off, set-down, and final positions. Thus, two extra interpolation points must be selected to provide enough boundary conditions for solving the unknown coefficients in the polynomial sequences. We can select these two extra knot points between the lift-off and set-down positions. It is not necessary to know these two locations exactly; we only require that the time intervals be known and that continuity of velocity and acceleration be satisfied at these two locations. Thus, the boundary conditions that this set of joint trajectory segment polynomials must satisfy are (1) position constraints at the initial, lift-off, set-down, and final positions and (2) continuity of velocity and acceleration at all the interpolation points. The boundary conditions

Table 4.3 Polynomial equations for 4-3-4 joint trajectory

First trajectory segment:

$$h_1(t) = \left(\delta_1 - v_0 t_1 - \frac{a_0 t_1^2}{2} - \sigma\right) t^4 + \sigma t^3 + \left(\frac{a_0 t_1^2}{2}\right) t^2 + (v_0 t_1) t + \theta_0$$

$$v_1 = \frac{\dot{h}_1(1)}{t_1} = \frac{4\delta_1}{t_1} - 3v_0 - a_0 t_1 - \frac{\sigma}{t_1}$$

$$a_1 = \frac{\ddot{h}_1(1)}{t_1^2} = \frac{12\delta_1}{t_1^2} - \frac{12v_0}{t_1} - 5a_0 - \frac{6\sigma}{t_1^2}$$

Second trajectory segment:

$$h_2(t) = \left(\delta_2 - v_1 t_2 - \frac{a_1 t_2^2}{2}\right) t^3 + \left(\frac{a_1 t_2^2}{2}\right) t^2 + (v_1 t_2) t + \theta_1$$

$$v_2 = \frac{\dot{h}_2(1)}{t_2} = \frac{3\delta_2}{t_2} - 2v_1 - \frac{a_1 t_2}{2}$$

$$a_2 = \frac{\ddot{h}_2(1)}{t_2^2} = \frac{6\delta_2}{t_2^2} - \frac{6v_1}{t_2} - 2a_1 t_2$$

Last trajectory segment:

$$h_n(t) = \left(9\delta_n - 4v_2 t_n - \frac{a_2 t_n^2}{2} - 5v_f t_n + \frac{a_f t_n^2}{2}\right) t^4 + \left(-8\delta_n + 5v_f t_n - \frac{a_f t_n^2}{2} + 3v_2 t_n\right) t^3 + \left(\frac{a_2 t_n^2}{2}\right) t^2 + (v_2 t_n) t + \theta_2$$

where $\sigma = f/g$ and

$$f = 2\delta_1 \left(4 + \frac{2t_n}{t_2} + \frac{2t_n}{t_1} + \frac{3t_2}{t_1}\right) - \frac{\delta_2 t_1}{t_2}\left(3 + \frac{t_n}{t_2}\right) + \frac{2\delta_n t_1}{t_n} - v_0 t_1 \left(6 + \frac{6t_2}{t_1} + \frac{4t_n}{t_1} + \frac{3t_n}{t_2}\right) - v_f t_1 - a_0 t_1 t_n \left(\frac{5}{3} + \frac{t_1}{t_2} + \frac{2t_1}{t_n} + \frac{5t_2}{2t_n}\right) + a_f t_1 t_n$$

$$g = \frac{t_n}{t_2} + \frac{2t_n}{t_1} + 2 + \frac{3t_2}{t_1}$$

Table 4.4 Polynomial equations for a 3-5-3 joint trajectory

First trajectory segment:

$$h_1(t) = \left(\delta_1 - v_0 t_1 - \frac{a_0 t_1^2}{2}\right) t^3 + \left(\frac{a_0 t_1^2}{2}\right) t^2 + (v_0 t_1) t + \theta_0$$

$$v_1 = \frac{\dot{h}_1(1)}{t_1} = \frac{3\delta_1}{t_1} - 2v_0 - \frac{a_0 t_1}{2}$$

$$a_1 = \frac{\ddot{h}_1(1)}{t_1^2} = \frac{6\delta_1}{t_1^2} - \frac{6v_0}{t_1} - 2a_0$$

Second trajectory segment:

$$h_2(t) = \left(6\delta_2 - 3v_1 t_2 - 3v_2 t_2 - \frac{a_1 t_2^2}{2} + \frac{a_2 t_2^2}{2}\right) t^5$$
$$+ \left(-15\delta_2 + 8v_1 t_2 + 7v_2 t_2 + \frac{3a_1 t_2^2}{2} - a_2 t_2^2\right) t^4$$
$$+ \left(10\delta_2 - 6v_1 t_2 - 4v_2 t_2 - \frac{3a_1 t_2^2}{2} + \frac{a_2 t_2^2}{2}\right) t^3 + \left(\frac{a_1 t_2^2}{2}\right) t^2$$
$$+ (v_1 t_2) t + \theta_1$$

$$v_2 = \frac{\dot{h}_2(1)}{t_2} = \frac{3\delta_n}{t_n} - 2v_f + \frac{a_f t_n}{2}$$

$$a_2 = \frac{\ddot{h}_2(1)}{t_2^2} = \frac{-6\delta_n}{t_n^2} + \frac{6v_f}{t_n} - 2a_f$$

Last trajectory segment:

$$h_n(t) = \left(\delta_n - v_f t_n + \frac{a_f t_n^2}{2}\right) t^3 + (-3\delta_n + 3v_f t_n - a_f t_n^2) t^2$$
$$+ \left(3\delta_n - 2v_f t_n + \frac{a_f t_n^2}{2}\right) t + \theta_2$$

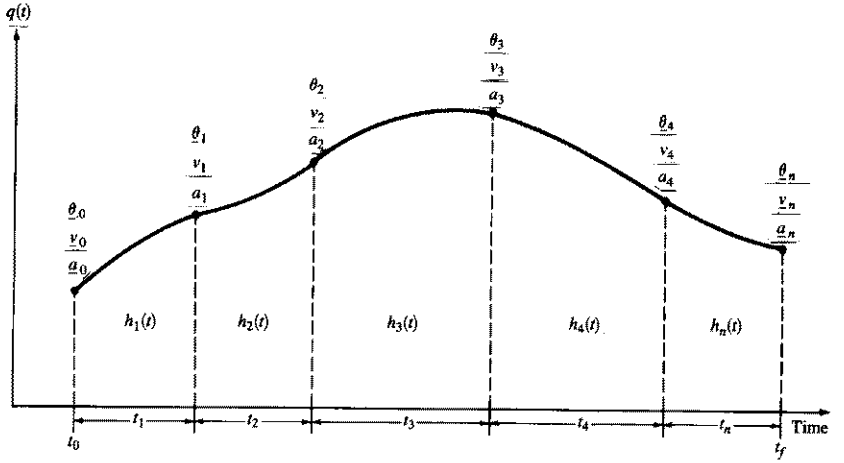

Figure 4.4 Boundary conditions for a 5-cubic joint trajectory.

for a five-cubic joint trajectory are shown in Fig. 4.4, where the underlined variables represent the known values before calculating the five-cubic polynomials.

The first and second derivatives of the polynomials with respect to real time are:

$$v_j(t) = \frac{\dot{h}_j(t)}{t_j} = \frac{3a_{j3}t^2 + 2a_{j2}t + a_{j1}}{t_j} \qquad j = 1, 2, 3, 4, n \tag{4.3-52}$$

and

$$a_j(t) = \frac{\ddot{h}_j(t)}{t_j^2} = \frac{6a_{j3}t + 2a_{j2}}{t_j^2} \qquad j = 1, 2, 3, 4, n \tag{4.3-53}$$

where t_j is the real time required to travel through the jth trajectory segment. Given the positions, velocities, and accelerations at the initial and final positions, the polynomial equations for the initial and final trajectory segments [$h_1(t)$ and $h_n(t)$] are completely determined. Once these polynomial equations are calculated, $h_2(t)$, $h_3(t)$, and $h_4(t)$ can be determined using the position constraints and continuity conditions.

For the first trajectory segment, the governing polynomial equation is

$$h_1(t) = a_{13}t^3 + a_{12}t^2 + a_{11}t + a_{10} \tag{4.3-54}$$

At $t = 0$, satisfying the boundary conditions at this position, we have

$$h_1(0) = a_{10} = \theta_0 \qquad \text{(given)} \tag{4.3-55}$$

$$v_0 \triangleq \frac{\dot{h}_1(0)}{t_1} = \frac{a_{11}}{t_1} \tag{4.3-56}$$

from which

$$a_{11} = v_0 t_1$$

and

$$a_0 \triangleq \frac{\ddot{h}_1(0)}{t_1^2} = \frac{2a_{12}}{t_1^2} \tag{4.3-57}$$

which yields

$$a_{12} = \frac{a_0 t_1^2}{2}$$

At $t = 1$, satisfying the position constraint at this position, we have

$$h_1(1) = a_{13} + \frac{a_0 t_1^2}{2} + v_0 t_1 + \theta_0 = \theta_1 \tag{4.3-58}$$

from which a_{13} is found to be

$$a_{13} = \delta_1 - v_0 t_1 - \frac{a_0 t_1^2}{2} \tag{4.3-59}$$

where $\delta_i = \theta_i - \theta_{i-1}$. Thus, the first trajectory segment polynomial is completely determined:

$$h_1(t) = \left(\delta_1 - v_0 t_1 - \frac{a_0 t_1^2}{2}\right) t^3 + \left(\frac{a_0 t_1^2}{2}\right) t^2 + (v_0 t_1) t + \theta_0 \tag{4.3-60}$$

With this polynomial equation, the velocity and acceleration at $t = 1$ are found to be

$$\frac{\dot{h}_1(1)}{t_1} \triangleq v_1 = \frac{3\delta_1 - (a_0 t_1^2)/2 - 2v_0 t_1}{t_1} = \frac{3\delta_1}{t_1} - 2v_0 - \frac{a_0 t_1}{2} \tag{4.3-61}$$

and

$$\frac{\ddot{h}_1(1)}{t_1^2} \triangleq a_1 = \frac{6\delta_1 - 2a_0 t_1^2 - 6v_0 t_1}{t_1^2} = \frac{6\delta_1}{t_1^2} - \frac{6v_0}{t_1} - 2a_0 \tag{4.3-62}$$

The velocity and acceleration must be continuous with the velocity and acceleration at the beginning of the next trajectory segment.

For the last trajectory segment, the polynomial equation is

$$h_n(t) = a_{n3} t^3 + a_{n2} t^2 + a_{n1} t + a_{n0} \tag{4.3-63}$$

At $t = 0$ and $t = 1$, satisfying the boundary conditions, we have

$$h_n(0) = a_{n0} = \theta_4 \qquad \text{(given)} \tag{4.3-64}$$

$$h_n(1) = a_{n3} + a_{n2} + a_{n1} + \theta_4 = \theta_f \tag{4.3-65}$$

$$\frac{\dot{h}_n(1)}{t_n} \triangleq v_f = \frac{3a_{n3} + 2a_{n2} + a_{n1}}{t_n} \tag{4.3-66}$$

and

$$\frac{\ddot{h}_n(1)}{t_n^2} \triangleq a_f = \frac{6a_{n3} + 2a_{n2}}{t_n^2} \tag{4.3-67}$$

Solving the above three equations for the unknown coefficients a_{n3}, a_{n2}, a_{n1}, we obtain

$$h_n(t) = \left(\delta_n - v_f t_n + \frac{a_f t_n^2}{2}\right) t^3 + (-3\delta_n + 3v_f t_n - a_f t_n^2)t^2 + \left(3\delta_n - 2v_f t_n + \frac{a_f t_n^2}{2}\right) t + \theta_4 \tag{4.3-68}$$

where $\delta_n = \theta_f - \theta_4$.

For the second trajectory segment, the equation is

$$h_2(t) = a_{23}t^3 + a_{22}t^2 + a_{21}t + a_{20} \tag{4.3-69}$$

At $t = 0$, satisfying the position constraint and the continuity of velocity and acceleration with the previous trajectory segment, we have

$$h_2(0) = a_{20} = \theta_1 \qquad \text{(given)} \tag{4.3-70}$$

$$v_1 = \frac{\dot{h}_2(0)}{t_2} = \frac{a_{21}}{t_2} = \frac{\dot{h}_1(1)}{t_1} \tag{4.3-71}$$

so that

$$a_{21} = v_1 t_2$$

and

$$a_1 = \frac{\ddot{h}_2(0)}{t_2^2} = \frac{2a_{22}}{t_2^2} = \frac{\ddot{h}_1(1)}{t_1^2} \tag{4.3-72}$$

which gives

$$a_{22} = \frac{a_1 t_2^2}{2}$$

With these unknowns determined, the polynomial equation becomes

$$h_2(t) = a_{23}t^3 + \left[\frac{a_1 t_2^2}{2}\right]t^2 + (v_1 t_2)t + \theta_1 \tag{4.3-73}$$

where $v_1 = \dfrac{3\delta_1}{t_1} - 2v_0 - \dfrac{a_0 t_1}{2}$ $\quad a_1 = \dfrac{6\delta_1}{t_1^2} - \dfrac{6v_0}{t_1} - 2a_0$

and a_{23} remains to be found. With this polynomial equation, at $t = 1$, we obtain the velocity and acceleration which must be continuous with the velocity and acceleration at the beginning of the next trajectory segment.

$$h_2(1) = \theta_2 = a_{23} + \frac{a_1 t_2^2}{2} + v_1 t_2 + \theta_1 \tag{4.3-74}$$

$$\frac{\dot{h}_2(1)}{t_2} = v_2 = \frac{3a_{23} + a_1 t_2^2 + v_1 t_2}{t_2} = v_1 + a_1 t_2 + \frac{3a_{23}}{t_2} \tag{4.3-75}$$

and $$\frac{\ddot{h}_2(1)}{t_2^2} = a_2 = \frac{6a_{23} + a_1 t_2^2}{t_2^2} = a_1 + \frac{6a_{23}}{t_2^2} \tag{4.3-76}$$

Note that θ_2, v_2, and a_2 all depend on the value of a_{23}.

For the third trajectory segment, the equation is

$$h_3(t) = a_{33}t^3 + a_{32}t^2 + a_{31}t + a_{30} \tag{4.3-77}$$

At $t = 0$, satisfying the continuity of velocity and acceleration with the previous trajectory segment, we have

$$h_3(0) = a_{30} = \theta_2 = a_{23} + \frac{a_1 t_2^2}{2} + v_1 t_2 + \theta_1 \tag{4.3-78}$$

$$v_2 \triangleq \frac{\dot{h}_3(0)}{t_3} = \frac{a_{31}}{t_3} = \frac{\dot{h}_2(1)}{t_2} \tag{4.3-79}$$

so that

$$a_{31} = v_2 t_3$$

and $$a_2 \triangleq \frac{\ddot{h}_3(0)}{t_3^2} = \frac{2a_{32}}{t_3^2} = \frac{\ddot{h}_2(1)}{t_2^2} \tag{4.3-80}$$

which yields

$$a_{32} = \frac{a_2 t_3^2}{2}$$

With these undetermined unknowns, the polynomial equation can be written as

$$h_3(t) = a_{33}t^3 + \left[\frac{a_2 t_3^2}{2}\right] t^2 + v_2 t_3 t + \theta_2 \tag{4.3-81}$$

At $t = 1$, we obtain the velocity and acceleration which are continuous with the velocity and acceleration at the beginning of the next trajectory segment.

$$h_3(1) = \theta_3 = \theta_2 + v_2 t_3 + \frac{a_2 t_3^2}{2} + a_{33} \tag{4.3-82}$$

$$\frac{\dot{h}_3(1)}{t_3} = v_3 = \frac{3a_{33} + a_2 t_3^2 + v_2 t_3}{t_3} = v_2 + a_2 t_3 + \frac{3a_{33}}{t_3} \tag{4.3-83}$$

and

$$\frac{\ddot{h}_3(1)}{t_3^2} = a_3 = \frac{6a_{33} + a_2 t_3^2}{t_3^2} = a_2 + \frac{6a_{33}}{t_3^2} \tag{4.3-84}$$

Note that θ_3, v_3, and a_3 all depend on a_{33} and implicitly depend on a_{23}.

For the fourth trajectory segment, the equation is

$$h_4(t) = a_{43}t^3 + a_{42}t^2 + a_{41}t + a_{40} \tag{4.3-85}$$

At $t = 0$, satisfying the position constraint and the continuity of velocity and acceleration with the previous trajectory segment, we have

$$h_4(0) = a_{40} = \theta_3 = \theta_2 + v_2 t_3 + \frac{a_2 t_3^2}{2} + a_{33} \tag{4.3-86}$$

$$v_3 = \frac{\dot{h}_4(0)}{t_4} = \frac{a_{41}}{t_4} = \frac{\dot{h}_3(1)}{t_3} \tag{4.3-87}$$

which gives

$$a_{41} = v_3 t_4$$

and

$$a_3 = \frac{\ddot{h}_4(0)}{t_4^2} = \frac{2a_{42}}{t_4^2} = \frac{\ddot{h}_3(1)}{t_3^2} \tag{4.3-88}$$

which yields

$$a_{42} = \frac{a_3 t_4^2}{2}$$

With these unknowns determined, the polynomial equation becomes

$$h_4(t) = a_{43}t^3 + \left(\frac{a_3 t_4^2}{2}\right)t^2 + (v_3 t_4)t + \theta_3 \tag{4.3-89}$$

where θ_3, v_3, and a_3 are given in Eqs. (4.3-82), (4.3-83), and (4.3-84), respectively, and a_{23}, a_{33}, a_{43} remain to be found. In order to completely determine the polynomial equations for the three middle trajectory segments, we need to determine the coefficients a_{23}, a_{33}, and a_{43}. This can be done by matching the condition at the endpoint of trajectory $h_4(t)$ with the initial point of $h_5(t)$:

$$h_4(1) = a_{43} + \frac{a_3 t_4^2}{2} + v_3 t_4 + \theta_3 = \theta_4 \tag{4.3-90}$$

$$\frac{\dot{h}_4(1)}{t_4} = \frac{3a_{43}}{t_4} + a_3 t_4 + v_3 = v_4 = \frac{3\delta_n}{t_n} - 2v_f + \frac{a_f t_n}{2} \tag{4.3-91}$$

and

$$\frac{\ddot{h}_4(1)}{t_4^2} = \frac{6a_{43}}{t_4^2} + a_3 = a_4 = \frac{-6\delta_n}{t_n^2} + \frac{6v_f}{t_n} - 2a_f \tag{4.3-92}$$

These three equations can be solved to determine the unknown coefficients a_{23}, a_{33}, and a_{43}. Solving for a_{23}, a_{33}, and a_{43}, the five-cubic polynomial equations are completely determined and they are listed below.

$$h_1(t) = \left(\delta_1 - v_0 t_1 - \frac{a_0 t_1^2}{2}\right)t^3 + \left(\frac{a_0 t_1^2}{2}\right)t^2 + (v_0 t_1)t + \theta_0 \tag{4.3-93}$$

$$v_1 = \frac{3\delta_1}{t_1} - 2v_0 - \frac{a_0 t_1}{2} \qquad a_1 = \frac{6\delta_1}{t_1^2} - \frac{6v_0}{t_1} - 2a_0 \tag{4.3-94}$$

$$h_2(t) = a_{23}t^3 + \left(\frac{a_1 t_2^2}{2}\right)t^2 + (v_1 t_2)t + \theta_1 \tag{4.3-95}$$

$$\theta_2 = a_{23} + \frac{a_1 t_2^2}{2} + v_1 t_2 + \theta_1 \tag{4.3-96}$$

$$v_2 = v_1 + a_1 t_2 + \frac{3a_{23}}{t_2} \qquad a_2 = a_1 + \frac{6a_{23}}{t_2^2} \tag{4.3-97}$$

$$h_3(t) = a_{33}t^3 + \left(\frac{a_2 t_3^2}{2}\right)t^2 + v_2 t_3 t + \theta_2 \tag{4.3-98}$$

$$\theta_3 = \theta_2 + v_2 t_3 + \frac{a_2 t_3^2}{2} + a_{33} \tag{4.3-99}$$

$$v_3 = v_2 + a_2 t_3 + \frac{3a_{33}}{t_3} \qquad a_3 = a_2 + \frac{6a_{33}}{t_3^2} \tag{4.3-100}$$

$$h_4(t) = a_{43} t^3 + \left(\frac{a_3 t_4^2}{2} \right) t^2 + (v_3 t_4) t + \theta_3 \tag{4.3-101}$$

$$h_n(t) = \left(\delta_n - v_f t_n + \frac{a_f t_n^2}{2} \right) t^3 + (-3\delta_n + 3v_f t_n - a_f t_n^2) t^2 + \left(3\delta_n - 2v_f t_n + \frac{a_f t_n^2}{2} \right) t + \theta_4 \tag{4.3-102}$$

$$v_4 = \frac{3\delta_n}{t_n} - 2v_f + \frac{a_f t_n}{2} \qquad a_4 = \frac{-6\delta_n}{t_n^2} + \frac{6v_f}{t_n} - 2a_f \tag{4.3-103}$$

$$a_{23} = t_2^2 \frac{x_1}{D} \qquad a_{33} = t_3^2 \frac{x_2}{D} \qquad a_{43} = t_4^2 \frac{x_3}{D} \tag{4.3-104}$$

with

$$x_1 = k_1(u - t_2) + k_2(t_4^2 - d) - k_3[(u - t_4)d + t_4^2(t_4 - t_2)] \tag{4.3-105}$$

$$x_2 = -k_1(u + t_3) + k_2(c - t_4^2) + k_3[(u - t_4)c + t_4^2(u - t_2)] \tag{4.3-106}$$

$$x_3 = k_1(u - t_4) + k_2(d - c) + k_3[(t_4 - t_2)c - d(u - t_2)] \tag{4.3-107}$$

$$D = u(u - t_2)(u - t_4) \tag{4.3-108}$$

$$u = t_2 + t_3 + t_4 \tag{4.3-109}$$

$$k_1 = \theta_4 - \theta_1 - v_1 u - a_1 \frac{u^2}{2} \tag{4.3-110}$$

$$k_2 = \frac{v_4 - v_1 - a_1 u - (a_4 - a_1)u/2}{3} \tag{4.3-111}$$

$$k_3 = \frac{a_4 - a_1}{6} \tag{4.3-112}$$

$$c = 3u^2 - 3ut_2 + t_2^2 \tag{4.3-113}$$

$$d = 3t_4^2 + 3t_3 t_4 + t_3^2 \tag{4.3-114}$$

So, it has been demonstrated that given the initial, the lift-off, the set-down, and the final positions, as well as the time to travel each trajectory (t_i), five-cubic polynomial equations can be uniquely determined to satisfy all the position con-

straints and continuity conditions. What we have just discussed is using a five-cubic polynomial to spline a joint trajectory with six interpolation points. A more general approach to finding the cubic polynomial for n interpolation points will be discussed in Sec. 4.4.3.

4.4 PLANNING OF CARTESIAN PATH TRAJECTORIES

In the last section, we described low-degree polynomial functions for generating joint-interpolated trajectory set points for the control of a manipulator. Although the manipulator joint coordinates fully specify the position and orientation of the manipulator hand, they are not suitable for specifying a goal task because most of the manipulator joint coordinates are not orthogonal and they do not separate position from orientation. For a more sophisticated robot system, programming languages are developed for controlling a manipulator to accomplish a task. In such systems, a task is usually specified as sequences of cartesian knot points through which the manipulator hand or end effector must pass. Thus, in describing the motions of the manipulator in a task, we are more concerned with the formalism of describing the target positions to which the manipulator hand has to move, as well as the space curve (or path) that it traverses.

Paul [1979] describes the design of manipulator cartesian paths made up of straight-line segments for the hand motion. The velocity and acceleration of the hand between these segments are controlled by converting them into the joint coordinates and smoothed by a quadratic interpolation routine. Taylor [1979] extended and refined Paul's method by using the dual-number quaternion representation to describe the location of the hand. Because of the properties of quaternions, transitions between the hand locations due to rotational operations require less computation, while the translational operations yield no advantage. We shall examine their approaches in designing straight-line cartesian paths in the next two sections.

4.4.1 Homogeneous Transformation Matrix Approach

In a programmable robotic system, the desired motion can be specified as sequences of cartesian knot points, each of which can be described in terms of homogeneous transformations relating the manipulator hand coordinate system to the workspace coordinate system. The corresponding joint coordinates at these cartesian knot points can be computed from the inverse kinematics solution routine and a quadratic polynominal can be used to smooth the two consecutive joint knot points in joint coordinates for control purposes. Thus, the manipulator hand is controlled to move along a straight line connected by these knot points. This technique has the advantage of enabling us to control the manipulator hand to track moving objects. Although the target positions are described by transforms, they do not specify how the manipulator hand is to be moved from one transform to another. Paul [1979] used a straight-line translation and two rotations to achieve

the motion between two consecutive cartesian knot points. The first rotation is about a unit vector **k** and serves to align the tool or end-effector along the desired approach angle and the second rotation aligns the orientation of the tool about the tool axis.

In general, the manipulator target positions can be expressed in the following fundamental matrix equation:

$$ {}^{0}\mathbf{T}_{6}\,{}^{6}\mathbf{T}_{\text{tool}} = {}^{0}\mathbf{C}_{\text{base}}(t)\,{}^{\text{base}}\mathbf{P}_{\text{obj}} \tag{4.4-1} $$

where

$ {}^{0}\mathbf{T}_{6} $ = 4 × 4 homogeneous transformation matrix describing the manipulator hand position and orientation with respect to the base coordinate frame.

$ {}^{6}\mathbf{T}_{\text{tool}} $ = 4 × 4 homogeneous transformation matrix describing the tool position and orientation with respect to the hand coordinate frame. It describes the tool endpoint whose motion is to be controlled.

$ {}^{0}\mathbf{C}_{\text{base}}(t) $ = 4×4 homogeneous transformation matrix function of time describing the working coordinate frame of the object with respect to the base coordinate frame.

$ {}^{\text{base}}\mathbf{P}_{\text{obj}} $ = 4×4 homogeneous transformation matrix describing the desired gripping position and orientation of the object for the end-effector with respect to the working coordinate frame.

If the $ {}^{6}\mathbf{T}_{\text{tool}} $ is combined with $ {}^{0}\mathbf{T}_{6} $ to form the arm matrix, then $ {}^{6}\mathbf{T}_{\text{tool}} $ is a 4 × 4 identity matrix and can be omitted. If the working coordinate system is the same as the base coordinate system of the manipulator, then $ {}^{0}\mathbf{C}_{\text{base}}(t) $ is a 4 × 4 identity matrix at all times.

Looking at Eq. (4.4-1), one can see that the left-hand-side matrices describe the gripping position and orientation of the manipulator, while the right-hand-side matrices describe the position and orientation of the feature of the object where we would like the manipulator's tool to grasp. Thus, we can solve for $ {}^{0}\mathbf{T}_{6} $ which describes the configuration of the manipulator for grasping the object in a correct and desired manner:

$$ {}^{0}\mathbf{T}_{6} = {}^{0}\mathbf{C}_{\text{base}}(t)\,{}^{\text{base}}\mathbf{P}_{\text{obj}}\,[{}^{6}\mathbf{T}_{\text{tool}}]^{-1} \tag{4.4-2} $$

If $ {}^{0}\mathbf{T}_{6} $ were evaluated at a sufficiently high rate and converted into corresponding joint angles, the manipulator could be servoed to follow the trajectory.

Utilizing Eq. (4.4-1), a sequence of N target positions defining a task can be expressed as

$$ \begin{aligned} {}^{0}\mathbf{T}_{6}\,({}^{6}\mathbf{T}_{\text{tool}})_{1} &= [{}^{0}\mathbf{C}_{\text{base}}(t)]_{1}\,({}^{\text{base}}\mathbf{P}_{\text{obj}})_{1} \\ {}^{0}\mathbf{T}_{6}\,({}^{6}\mathbf{T}_{\text{tool}})_{2} &= [{}^{0}\mathbf{C}_{\text{base}}(t)]_{2}\,({}^{\text{base}}\mathbf{P}_{\text{obj}})_{2} \\ &\vdots \\ {}^{0}\mathbf{T}_{6}\,({}^{6}\mathbf{T}_{\text{tool}})_{N} &= [{}^{0}\mathbf{C}_{\text{base}}(t)]_{N}\,({}^{\text{base}}\mathbf{P}_{\text{obj}})_{N} \end{aligned} \tag{4.4-3} $$

Simplifying the notation of superscript and subscript in the above equation, we have

$$\begin{aligned} \mathbf{T}_6\,{}^{\text{tool}}\mathbf{T}_1 &= \mathbf{C}_1(t)\,\mathbf{P}_1 \\ \mathbf{T}_6\,{}^{\text{tool}}\mathbf{T}_2 &= \mathbf{C}_2(t)\,\mathbf{P}_2 \\ &\vdots \\ \mathbf{T}_6\,{}^{\text{tool}}\mathbf{T}_N &= \mathbf{C}_N(t)\,\mathbf{P}_N \end{aligned} \tag{4.4-4}$$

From the positions defined by $\mathbf{C}_i(t)\,\mathbf{P}_i$ we can obtain the distance between consecutive points, and if we are further given linear and angular velocities, we can obtain the time requested T_i to move from position i to position $i+1$. Since tools and moving coordinate systems are specified at positions with respect to the base coordinate system, moving from one position to the next is best done by specifying both positions and tools with respect to the destination position. This has the advantage that the tool appears to be at rest from the moving coordinate system. In order to do this, we need to redefine the present position and tools with respect to the subsequent coordinate system. This can easily be done by redefining the $\mathbf{P}_i$ transform using a two subscript notation as $\mathbf{P}_{ij}$ which indicates the position $\mathbf{P}_i$ expressed with respect to the jth coordinate system. Thus, if the manipulator needs to be controlled from position 1 to position 2, then at position 1, expressing it with respect to its own coordinate system, we have

$$\mathbf{T}_6\,{}^{\text{tool}}\mathbf{T}_1 = \mathbf{C}_1(t)\,\mathbf{P}_{11} \tag{4.4-5}$$

and expressing it with respect to the position 2 coordinate system, we have

$$\mathbf{T}_6\,{}^{\text{tool}}\mathbf{T}_2 = \mathbf{C}_2(t)\,\mathbf{P}_{12} \tag{4.4-6}$$

We can now obtain $\mathbf{P}_{12}$ from these equations:

$$\mathbf{P}_{12} = \mathbf{C}_2^{-1}(t)\,\mathbf{C}_1(t)\,\mathbf{P}_{11}\,({}^{\text{tool}}\mathbf{T}_1)^{-1}\,{}^{\text{tool}}\mathbf{T}_2 \tag{4.4-7}$$

The purpose of the above equation is to find $\mathbf{P}_{12}$ given $\mathbf{P}_{11}$. Thus, the motion between any two consecutive positions i and $i+1$ can be stated as a motion from

$$\mathbf{T}_6 = \mathbf{C}_{i+1}(t)\,\mathbf{P}_{i,i+1}\,({}^{\text{tool}}\mathbf{T}_{i+1})^{-1} \tag{4.4-8}$$

to

$$\mathbf{T}_6 = \mathbf{C}_{i+1}(t)\,\mathbf{P}_{i+1,i+1}\,({}^{\text{tool}}\mathbf{T}_{i+1})^{-1} \tag{4.4-9}$$

where $\mathbf{P}_{i,i+1}$ and $\mathbf{P}_{i+1,i+1}$ represent transforms, as discussed above. Paul [1979] used a simple way to control the manipulator hand moving from one transform to the other. The scheme involves a translation and a rotation about a fixed axis in space coupled with a second rotation about the tool axis to produce controlled linear and angular velocity motion of the manipulator hand. The first rotation serves to align the tool in the required approach direction and the second rotation serves to align the orientation vector of the tool about the tool axis.

The motion from position i to position $i+1$ can be expressed in terms of a "drive" transform, $\mathbf{D}(\lambda)$, which is a function of a normalized time λ, as

$$\mathbf{T}_6(\lambda) = \mathbf{C}_{i+1}(\lambda)\,\mathbf{P}_{i,i+1}\,\mathbf{D}(\lambda)\,({}^{\text{tool}}\mathbf{T}_{i+1})^{-1} \tag{4.4-10}$$

where

$\lambda = \dfrac{t}{T}$, $\lambda \in [0, 1]$

t = real time since the beginning of the motion

T = total time for the traversal of this segment

At position i, the real time is zero, λ is zero, $\mathbf{D}(0)$ is a 4×4 identity matrix, and

$$\mathbf{P}_{i+1,i+1} = \mathbf{P}_{i,i+1}\,\mathbf{D}(1) \tag{4.4-11}$$

which gives

$$\mathbf{D}(1) = (\mathbf{P}_{i,i+1})^{-1}\,\mathbf{P}_{i+1,i+1} \tag{4.4-12}$$

Expressing the positions i and $i+1$ in their respective homogeneous transform matrices, we have

$$\mathbf{P}_{i,i+1} \triangleq \mathbf{A} = \begin{bmatrix} \mathbf{n}_A & \mathbf{s}_A & \mathbf{a}_A & \mathbf{p}_A \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x^A & s_x^A & a_x^A & p_x^A \\ n_y^A & s_y^A & a_y^A & p_y^A \\ n_z^A & s_z^A & a_z^A & p_z^A \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-13}$$

and

$$\mathbf{P}_{i+1,i+1} \triangleq \mathbf{B} = \begin{bmatrix} \mathbf{n}_B & \mathbf{s}_B & \mathbf{a}_B & \mathbf{p}_B \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x^B & s_x^B & a_x^B & p_x^B \\ n_y^B & s_y^B & a_y^B & p_y^B \\ n_z^B & s_z^B & a_z^B & p_z^B \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-14}$$

Using Eq. (2.2-27) to invert $\mathbf{P}_{i,i+1}$ and multiply with $\mathbf{P}_{i+1,i+1}$, we obtain

$$\mathbf{D}(1) = \begin{bmatrix} \mathbf{n}_A \cdot \mathbf{n}_B & \mathbf{n}_A \cdot \mathbf{s}_B & \mathbf{n}_A \cdot \mathbf{a}_B & \mathbf{n}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \\ \mathbf{s}_A \cdot \mathbf{n}_B & \mathbf{s}_A \cdot \mathbf{s}_B & \mathbf{s}_A \cdot \mathbf{a}_B & \mathbf{s}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \\ \mathbf{a}_A \cdot \mathbf{n}_B & \mathbf{a}_A \cdot \mathbf{s}_B & \mathbf{a}_A \cdot \mathbf{a}_B & \mathbf{a}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-15}$$

where the dot indicates the scalar product of two vectors.

If the drive function consists of a translational motion and two rotational motions, then both the translation and the rotations will be directly proportional to λ. If λ varies linearly with the time, then the resultant motion represented by

$\mathbf{D}(\lambda)$ will correspond to a constant linear velocity and two angular velocities. The translational motion can be represented by a homogeneous transformation matrix $\mathbf{L}(\lambda)$ and the motion will be along the straight line joining $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$. The first rotational motion can be represented by a homogeneous transformation matrix $\mathbf{R}_A(\lambda)$ and it serves to rotate the approach vector from $\mathbf{P}_i$ to the approach vector at $\mathbf{P}_{i+1}$. The second rotational motion represented by $\mathbf{R}_B(\lambda)$ serves to rotate the orientation vector from $\mathbf{P}_i$ into the orientation vector at $\mathbf{P}_{i+1}$ about the tool axis. Thus, the drive function can be represented as

$$\mathbf{D}(\lambda) = \mathbf{L}(\lambda)\,\mathbf{R}_A(\lambda)\,\mathbf{R}_B(\lambda) \tag{4.4-16}$$

where

$$\mathbf{L}(\lambda) = \begin{bmatrix} 1 & 0 & 0 & \lambda x \\ 0 & 1 & 0 & \lambda y \\ 0 & 0 & 1 & \lambda z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-17}$$

$$\mathbf{R}_A(\lambda) = \begin{bmatrix} S^2\psi V(\lambda\theta) + C(\lambda\theta) & -S\psi C\psi V(\lambda\theta) & C\psi S(\lambda\theta) & 0 \\ -S\psi C\psi V(\lambda\theta) & C^2\psi V(\lambda\theta) + C(\lambda\theta) & S\psi S(\lambda\theta) & 0 \\ -C\psi S(\lambda\theta) & -S\psi S(\lambda\theta) & C(\lambda\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-18}$$

$$\mathbf{R}_B(\lambda) = \begin{bmatrix} C(\lambda\phi) & -S(\lambda\phi) & 0 & 0 \\ S(\lambda\phi) & C(\lambda\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-19}$$

where

$$V(\lambda\theta) = \text{Versine}(\lambda\theta) = 1 - \cos(\lambda\theta) \tag{4.4-20}$$

$$C(\lambda\theta) = \cos(\lambda\theta) \qquad S(\lambda\theta) = \sin(\lambda\theta)$$

$$C(\lambda\phi) = \cos(\lambda\phi) \qquad S(\lambda\phi) = \sin(\lambda\phi)$$

and $\lambda \in [0, 1]$. The rotation matrix $\mathbf{R}_A(\lambda)$ indicates a rotation of an angle θ about the orientation vector of $\mathbf{P}_i$ which is rotated an angle of ψ about the approach vector. $\mathbf{R}_B(\lambda)$ represents a rotation of ϕ about the approach vector of the tool at $\mathbf{P}_{i+1}$.

Multiplying the matrices in Eqs. (4.4-17) to (4.4-19) together, we have

$$\mathbf{D}(\lambda) = \begin{bmatrix} \mathbf{dn} & \mathbf{do} & \mathbf{da} & \mathbf{dp} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4-21}$$

where

$$\mathbf{do} = \begin{bmatrix} -S(\lambda\phi)[S^2\psi V(\lambda\theta) + C(\lambda\theta)] + C(\lambda\phi)[-S\psi C\psi V(\lambda\theta)] \\ -S(\lambda\phi)[-S\psi C\psi V(\lambda\theta)] + C(\lambda\theta)[C^2\psi V(\lambda\theta) + C(\lambda\theta)] \\ -S(\lambda\phi)[-C\psi S(\lambda\theta)] + C(\lambda\theta)[-S\psi S(\lambda\theta)] \end{bmatrix}$$

$$\mathbf{da} = \begin{bmatrix} C\psi S(\lambda\theta) \\ S\psi S(\lambda\theta) \\ C(\lambda\theta) \end{bmatrix} \qquad \mathbf{dp} = \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda z \end{bmatrix}$$

and

$$\mathbf{dn} = \mathbf{do} \times \mathbf{da}$$

Using the inverse transform technique on Eq. (4.4-16), we may solve for x, y, z by postmultiplying Eq. (4.4-16) by $\mathbf{R}_B^{-1}(\lambda)\,\mathbf{R}_A^{-1}(\lambda)$ and equating the elements of the position vector,

$$\begin{aligned} x &= \mathbf{n}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \\ y &= \mathbf{s}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \\ z &= \mathbf{a}_A \cdot (\mathbf{p}_B - \mathbf{p}_A) \end{aligned} \tag{4.4-22}$$

By postmultiplying both sides of Eq. (4.4-16) by $\mathbf{R}_B^{-1}(\lambda)$ and then premultiplying by $\mathbf{L}^{-1}(\lambda)$, we can solve for θ and ψ by equating the elements of the third column with the elements of the third column from Eq. (4.4-16),

$$\psi = \tan^{-1}\left(\frac{\mathbf{s}_A \cdot \mathbf{a}_B}{\mathbf{n}_A \cdot \mathbf{a}_B}\right) \qquad -\pi \leqslant \psi < \pi \tag{4.4-23}$$

$$\theta = \tan^{-1}\left\{\frac{[(\mathbf{n}_A \cdot \mathbf{a}_B)^2 + (\mathbf{s}_A \cdot \mathbf{a}_B)^2]^{1/2}}{\mathbf{a}_A \cdot \mathbf{a}_B}\right\} \qquad 0 \leqslant \theta \leqslant \pi \tag{4.4-24}$$

To find ϕ, we premultiply both sides of Eq. (4.4-16) by $\mathbf{L}^{-1}(\lambda)$ and then $\mathbf{R}_A^{-1}(\lambda)$ and equate the elements to obtain

$$\begin{aligned} S\phi = {} & -S\psi C\psi V(\lambda\theta)(\mathbf{n}_A \cdot \mathbf{n}_B) + [C^2\psi V(\lambda\theta) + C(\lambda\theta)](\mathbf{s}_A \cdot \mathbf{n}_B) \\ & - S\psi S(\lambda\theta)(\mathbf{a}_A \cdot \mathbf{n}_B) \end{aligned} \tag{4.4-25}$$

and

$$\begin{aligned} C\phi = {} & -S\psi C\psi V(\lambda\theta)(\mathbf{n}_A \cdot \mathbf{s}_B) + [C^2\psi V(\lambda\theta) + C(\lambda\theta)](\mathbf{s}_A \cdot \mathbf{s}_B) \\ & - S\psi S(\lambda\theta)(\mathbf{a}_A \cdot \mathbf{s}_B) \end{aligned} \tag{4.4-26}$$

then

$$\phi = \tan^{-1}\left(\frac{S\phi}{C\phi}\right) \qquad -\pi \leqslant \phi < \pi \tag{4.4-27}$$

Transition Between Two Path Segments. Quite often, a manipulator has to move on connected straight-line segments to satisfy a task motion specification or to avoid obstacles. In order to avoid discontinuity of velocity at the endpoint of each segment, we must accelerate or decelerate the motion from one segment to another segment. This can be done by initiating a change in velocity τ unit of time before the manipulator reaches an endpoint and maintaining the acceleration constant until τ unit of time into the new motion segment (see Fig. 4.5). If the acceleration for each variable is maintained at a constant value from time $-\tau$ to τ, then the acceleration necessary to change both the position and velocity is

$$\ddot{\mathbf{q}}(t) = \frac{1}{2\tau^2}\left(\Delta\mathbf{C}\frac{\tau}{T} + \Delta\mathbf{B}\right) \tag{4.4-28}$$

where $-\tau < t < T$ and

$$\ddot{\mathbf{q}}(t) = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} \qquad \Delta\mathbf{C} = \begin{bmatrix} x_{BC} \\ y_{BC} \\ z_{BC} \\ \theta_{BC} \\ \phi_{BC} \end{bmatrix} \qquad \Delta\mathbf{B} = \begin{bmatrix} x_{BA} \\ y_{BA} \\ z_{BA} \\ \theta_{BA} \\ \phi_{BA} \end{bmatrix}$$

where $\Delta\mathbf{C}$ and $\Delta\mathbf{B}$ are vectors whose elements are cartesian distances and angles from points B to C and from points B to A, respectively.

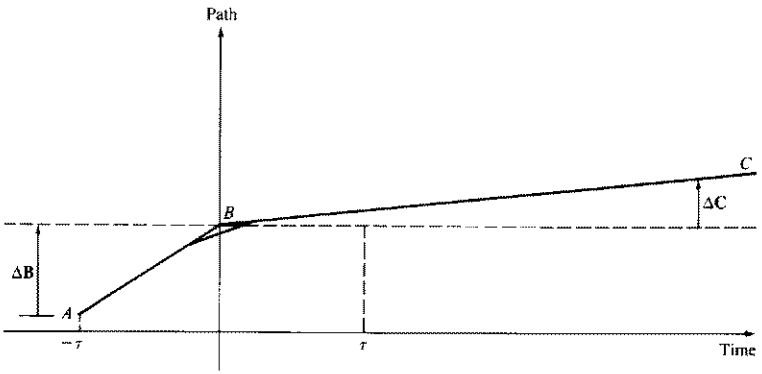

Figure 4.5 Straight line transition between two segments.

From Eq. (4.4-28), the velocity and position for $-\tau < t < \tau$ are given by

$$\dot{\mathbf{q}}(t) = \frac{1}{\tau}\left[\Delta\mathbf{C}\frac{\tau}{T} + \Delta\mathbf{B}\right]\lambda - \frac{\Delta\mathbf{B}}{\tau} \tag{4.4-29}$$

$$\mathbf{q}(t) = \left[\left[\Delta\mathbf{C}\frac{\tau}{T} + \Delta\mathbf{B}\right]\lambda - 2\Delta\mathbf{B}\right]\lambda + \Delta\mathbf{B} \tag{4.4-30}$$

where

$$\mathbf{q}(t) = \begin{bmatrix} x \\ y \\ z \\ \theta \\ \phi \end{bmatrix} \qquad \dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} \qquad \lambda \triangleq \frac{t + \tau}{2\tau} \tag{4.4-31}$$

For $\tau < t < T$, the motion is described by

$$\mathbf{q} = \Delta\mathbf{C}\lambda \qquad \dot{\mathbf{q}} = 0 \tag{4.4-32}$$

where

$$\lambda \triangleq \frac{t}{T}$$

It is noted that, as before, λ represents normalized time in the range [0, 1]. The reader should bear in mind, however, that the normalization factors are usually different for different time intervals.

For the motion from A to B and to C, we define a ψ as a linear interpolation between the motions for $-\tau < t < \tau$ as

$$\psi = (\psi_{BC} - \psi_{AB})\lambda + \psi_{AB} \tag{4.4-33}$$

where ψ_{AB} and ψ_{BC} are defined for the motion from A to B and from B to C, respectively, as in Eq. (4.4-23). Thus, ψ will change from ψ_{AB} to ψ_{BC}.

In summary, to move from a position $\mathbf{P}_i$ to a position $\mathbf{P}_{i+1}$, the drive function $\mathbf{D}(\lambda)$ is computed using Eqs. (4.4-16) to (4.4-27); then $\mathbf{T}_6(\lambda)$ can be evaluated by Eq. (4.4-10) and the corresponding joint values can be calculated from the inverse kinematics routine. If necessary, quadratic polynominal functions can be used to interpolate between the points obtained from the inverse kinematics routine.

Example: A robot is commanded to move in straight line motion to place a bolt into one the holes in the bracket shown in Fig. 4.6. Write down all the necessary matrix equations relationships as discussed above so that the robot can move along the dotted lines and complete the task. You may use symbols to indicate intermediate positions along the straight line paths.

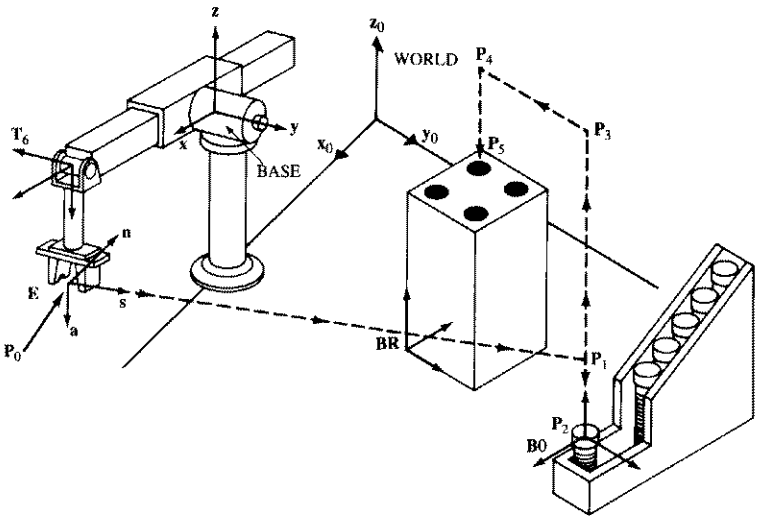

Figure 4.6 Figure for the example.

SOLUTION: Let $\mathbf{P}_i$ be the cartesian knot points that the manipulator hand must traverse ($i = 0, 1, 2, 3, 4, 5$). Then the governing matrix equations are:

$$\text{At } \mathbf{P}_0: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{INIT}]\,[\mathbf{P}_0] \tag{4.4-34}$$

$$\text{At } \mathbf{P}_1: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{B0}]\,[\mathbf{P}_1] \tag{4.4-35}$$

$$\text{At } \mathbf{P}_2: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{B0}]\,[\mathbf{P}_2] \tag{4.4-36}$$

$$\text{At } \mathbf{P}_3: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{B0}]\,[\mathbf{P}_3] \tag{4.4-37}$$

$$\text{At } \mathbf{P}_4: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{BR}]\,[\mathbf{P}_4] \tag{4.4-38}$$

$$\text{At } \mathbf{P}_5: \quad [\mathbf{BASE}]\,[\mathbf{T6}]\,[\mathbf{E}] = [\mathbf{BR}]\,[\mathbf{P}_5] \tag{4.4-39}$$

where **[WORLD]**, **[BASE]**, **[INIT]**, **[B0]**, **[BR]**, **[T6]**, **[E]**, $[\mathbf{P}_0]$, $[\mathbf{P}_1]$, $[\mathbf{P}_2]$, $[\mathbf{P}_3]$, $[\mathbf{P}_4]$, and $[\mathbf{P}_5]$ are 4×4 coordinate matrices. **[BASE]**, **[INIT]**, **[B0]**, and **[BR]** are expressed with respect to **[WORLD]**; **[T6]** is expressed with respect to **[BASE]**; **[E]** is expressed with respect to **[T6]**; $[\mathbf{P}_0]$ is expressed with respect to **[INIT]**; $[\mathbf{P}_1]$, $[\mathbf{P}_2]$, and $[\mathbf{P}_3]$ are expressed with respect to **[B0]**; and $[\mathbf{P}_4]$ and $[\mathbf{P}_5]$ are expressed with respect to **[BR]**.

To move from location $\mathbf{P}_0$ to location $\mathbf{P}_1$, we use the double subscript to describe Eq. (4.4-34) with respect to $\mathbf{P}_1$ coordinate frame. From Eq. (4.4-34), we have

$$[\mathbf{T6}] = [\mathbf{BASE}]^{-1}\,[\mathbf{INIT}]\,[\mathbf{P}_{00}]\,[\mathbf{E}]^{-1} \tag{4.4-40}$$

and expressing it with respect to $\mathbf{P}_1$, we have

$$[\mathbf{T6}] = [\mathbf{BASE}]^{-1}\,[\mathbf{B0}]\,[\mathbf{P}_{01}]\,[\mathbf{E}]^{-1} \tag{4.4-41}$$

Equating Eqs. (4.4-40) and (4.4-41), we have

$$[\mathbf{P}_{01}] = [\mathbf{B0}]^{-1}\,[\mathbf{INIT}]\,[\mathbf{P}_{00}] \tag{4.4-42}$$

Thus, moving from location $\mathbf{P}_0$ to location $\mathbf{P}_1$ in a straight-line motion means that the manipulator hand must change configuration from

$$[\mathbf{T6}] = [\mathbf{BASE}]^{-1}\,[\mathbf{B0}]\,[\mathbf{P}_{01}]\,[\mathbf{E}]^{-1} \tag{4.4-43a}$$

to

$$[\mathbf{T6}] = [\mathbf{BASE}]^{-1}\,[\mathbf{B0}]\,[\mathbf{P}_{11}]\,[\mathbf{E}]^{-1} \tag{4.4-43b}$$

Moving from locations $\mathbf{P}_i$ to $\mathbf{P}_{i+1}$, $i = 1, 2, 3, 4$, can be solved in the same manner. □

4.4.2 Planning Straight-Line Trajectories Using Quaternions

Paul's straight-line motion trajectory scheme uses the homogeneous transformation matrix approach to represent target position. This representation is easy to understand and use. However, the matrices are moderately expensive to store and computations on them require more operations than for some other representations. Furthermore, matrix representation for rotations is highly redundant, and this may lead to numerical inconsistencies. Taylor [1979] noted that using a quaternion to represent rotation will make the motion more uniform and efficient. He proposed two approaches to the problem of planning straight-line motion between knot points. The first approach, called *cartesian path control*, is a refinement of Paul's technique but using a quaternion representation for rotations. This method is simple and provides more uniform rotational motion. However, it requires considerable real time computation and is vulnerable to degenerate manipulator configurations. The second approach, called *bounded deviation joint path*, requires a motion planning phase which selects enough knot points so that the manipulator can be controlled by linear interpolation of joint values without allowing the manipulator hand to deviate more than a prespecified amount from the straight-line path. This approach greatly reduces the amount of computation that must be done at every sample interval.

Quaternion Representation. The quaternion concept has been successfully applied to the analysis of spatial mechanisms for the last several decades. We shall use quaternions to facilitate the representation of the orientation of a manipulator hand for planning a straight-line trajectory. A quaternion is a quadruple of ordered real

numbers, s, a, b, c, associated, respectively, with four units: the real number $+1$, and three other units $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$, having cyclical permutation:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1$$

$$\mathbf{ij} = \mathbf{k} \qquad \mathbf{jk} = \mathbf{i} \qquad \mathbf{ki} = \mathbf{j}$$

$$\mathbf{ji} = -\mathbf{k} \qquad \mathbf{kj} = -\mathbf{i} \qquad \mathbf{ik} = -\mathbf{j}$$

The units $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ of a quaternion may be interpreted as the three basis vectors of a cartesian set of axes. Thus, a quaternion Q may be written as a scalar part s and a vector part v:

$$Q = [s + \mathbf{v}] = s + a\mathbf{i} + b\mathbf{j} + c\mathbf{k} = (s, a, b, c) \tag{4.4-44}$$

The following properties of quaternion algebra are basic:

Scalar part of Q:	s
Vector part of Q:	$a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$
Conjugate of Q:	$s - (a\mathbf{i} + b\mathbf{j} + c\mathbf{k})$
Norm of Q:	$s^2 + a^2 + b^2 + c^2$
Reciprocal of Q:	$\dfrac{s - (a\mathbf{i} + b\mathbf{j} + c\mathbf{k})}{s^2 + a^2 + b^2 + c^2}$
Unit quaternion:	$s + a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$, where $s^2 + a^2 + b^2 + c^2 = 1$

It is important to note that quaternions include the real numbers $(s, 0, 0, 0)$ with a single unit 1, the complex numbers $(s, a, 0, 0)$ with two units 1 and $\mathbf{i}$, and the vectors $(0, a, b, c)$ in a three-dimensional space. The addition (subtraction) of two quaternions equals adding (subtracting) corresponding elements in the quadruples. The multiplication of two quaternions can be written as

$$\begin{aligned} Q_1 Q_2 &= (s_1 + a_1\mathbf{i} + b_1\mathbf{j} + c_1\mathbf{k})(s_2 + a_2\mathbf{i} + b_2\mathbf{j} + c_2\mathbf{k}) \\ &= (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 + s_2\mathbf{v}_1 + s_1\mathbf{v}_2 + \mathbf{v}_1 \times \mathbf{v}_2) \end{aligned} \tag{4.4-45}$$

and is obtained by distributing the terms on the right as in ordinary algebra, except that the order of the units must be preserved. In general, the product of two vectors in three-dimensional space, expressed as quaternions, is not a vector but a quaternion. That is, $Q_1 = [0 + \mathbf{v}_1] = (0, a_1, b_1, c_1)$ and $Q_2 = [0 + \mathbf{v}_2] = (0, a_2, b_2, c_2)$ and from Eq. (4.4-45),

$$Q_1 Q_2 = -\mathbf{v}_1 \cdot \mathbf{v}_2 + \mathbf{v}_1 \times \mathbf{v}_2$$

With the aid of quaternion algebra, finite rotations in space may be dealt with in a simple and efficient manner. If we use the notation

$$S = \sin\left(\frac{\theta}{2}\right) \qquad \text{and} \qquad C = \cos\left(\frac{\theta}{2}\right)$$

then we can represent a rotation Rot $(\mathbf{n}, \theta)$ of angle θ about an axis $\mathbf{n}$ by a quaternion,

$$\text{Rot}(\mathbf{n}, \theta) = \left[\cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\mathbf{n}\right] \tag{4.4-46}$$

Example: A rotation of 90° about $\mathbf{k}$ followed by a rotation of 90° about $\mathbf{j}$ is represented by the quaternion product

$$(\cos 45° + \mathbf{j}\sin 45°)(\cos 45° + \mathbf{k}\sin 45°) = (\tfrac{1}{2} + \mathbf{j}\tfrac{1}{2} + \mathbf{k}\tfrac{1}{2} + \mathbf{i}\tfrac{1}{2})$$

$$= \left(\tfrac{1}{2} + \frac{\mathbf{i}+\mathbf{j}+\mathbf{k}}{\sqrt{3}}\frac{\sqrt{3}}{2}\right)$$

$$= \left(\cos 60° + \sin 60°\frac{\mathbf{i}+\mathbf{j}+\mathbf{k}}{\sqrt{3}}\right)$$

$$= \text{Rot}\left(\frac{\mathbf{i}+\mathbf{j}+\mathbf{k}}{\sqrt{3}}, 120°\right)$$

The resultant rotation is a rotation of 120° about an axis equally inclined to the $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ axes. Note that we could represent the rotations about the $\mathbf{j}$ and $\mathbf{k}$ axes using the rotation matrices discussed in Chap. 2. However, the quaternion gives a much simpler representation. Thus, one can change the representation for rotations from quaternion to matrix or vice versa. □

For the remainder of this section, finite rotations will be represented in quaternion as Rot $(\mathbf{n}, \theta) = [\cos(\theta/2) + \sin(\theta/2)\mathbf{n}]$ for a rotation of angle θ about an axis $\mathbf{n}$. Table 4.5 lists the computational requirements of some common rotation operations, using quaternion and matrix representations.

Table 4.5 Computational requirements using quaternions and matrices

Operation	Quaternion representation	Matrix representation
$\mathbf{R}_1\mathbf{R}_2$	9 adds, 16 multiplies	15 adds, 24 multiplies
$\mathbf{R}v$	12 adds, 22 multiplies	6 adds, 9 multiplies
$\mathbf{R} \rightarrow \text{Rot}(\mathbf{n}, \theta)$	4 multiplies, 1 square root, 1 arctangent	8 adds, 10 multiplies, 2 square roots, 1 arctangent

Cartesian Path Control Scheme. It is required to move the manipulator's hand coordinate frame along a straight-line path between two knot points specified by $\mathbf{F}_0$ and $\mathbf{F}_1$ in time T, where each coordinate frame is represented by a homogeneous transformation matrix,

$$\mathbf{F}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{p}_i \\ 0 & 1 \end{bmatrix}$$

The motion along the path consists of translation of the tool frame's origin from $\mathbf{p}_0$ to $\mathbf{p}_1$ coupled with rotation of the tool frame orientation part from $\mathbf{R}_0$ to $\mathbf{R}_1$. Let $\lambda(t)$ be the remaining fraction of the motion still to be traversed at time t. Then for uniform motion, we have

$$\lambda(t) = \frac{T - t}{T} \tag{4.4-47}$$

where T is the total time needed to traverse the segment and t is time starting from the beginning of the segment traversal. The tool frame's position and orientation at time t are given, respectively, by

$$\mathbf{p}(t) = \mathbf{p}_1 - \lambda(t)(\mathbf{p}_1 - \mathbf{p}_0) \tag{4.4-48}$$

$$\mathbf{R}(t) = \mathbf{R}_1 \,\mathrm{Rot}\,[\mathbf{n}, -\theta\lambda(t)] \tag{4.4-49}$$

where $\mathrm{Rot}(\mathbf{n}, \theta)$ is a rotation by θ about an axis $\mathbf{n}$ to reorient $\mathbf{R}_0$ into $\mathbf{R}_1$,

$$\mathrm{Rot}\,(\mathbf{n}, \theta) = \mathbf{R}_0^{-1}\mathbf{R}_1 \tag{4.4-50}$$

where $\mathrm{Rot}\,(\mathbf{n}, \theta)$ represents the resultant rotation of $\mathbf{R}_0^{-1}\mathbf{R}_1$ in quaternion form. It is worth noting that $\mathbf{p}_1 - \mathbf{p}_0$ in Eq. (4.4-48) and $\mathbf{n}$ and θ in Eq. (4.4-49) need to be evaluated only once per segment if the frame $\mathbf{F}_1$ is fixed. On the other hand, if the destination point is changing, then $\mathbf{F}_1$ will be changing too. In this case, $\mathbf{p}_1 - \mathbf{p}_0$, $\mathbf{n}$, and θ should be evaluated per step. This can be accomplished by the pursuit formulation as described by Taylor [1979].

If the manipulator hand is required to move from one segment to another while maintaining constant acceleration, then it must accelerate or decelerate from one segment to the next. In order to accomplish this, the transition must start τ time before the manipulator reaches the intersection of the two segments and complete the transition to the new segment at time τ after the intersection with the new segment has been reached. From this requirement, the boundary conditions for the segment transition are

$$\mathbf{p}(T_1 - \tau) = \mathbf{p}_1 - \frac{\tau\Delta\mathbf{p}_1}{T_1} \tag{4.4-51}$$

$$\mathbf{p}(T_1 + \tau) = \mathbf{p}_1 + \frac{\tau \Delta \mathbf{p}_2}{T_2} \tag{4.4-52}$$

$$\frac{d}{dt}\mathbf{p}(t)\Big|_{t=T_1-\tau} = \frac{\Delta \mathbf{p}_1}{T_1} \tag{4.4-53}$$

$$\frac{d}{dt}\mathbf{p}(t)\Big|_{t=T_1+\tau} = \frac{\Delta \mathbf{p}_2}{T_2} \tag{4.4-54}$$

where $\Delta \mathbf{p}_1 = \mathbf{p}_1 - \mathbf{p}_2$, $\Delta \mathbf{p}_2 = \mathbf{p}_2 - \mathbf{p}_1$, and T_1 and T_2 are the traversal times for the two segments. If we apply a constant acceleration to the transition,

$$\frac{d^2}{dt^2}\mathbf{p}(t) = \mathbf{a}_p \tag{4.4-55}$$

then integrating the above equation twice and applying the boundary conditions gives the position equation of the tool frame,

$$\mathbf{p}(t') = \mathbf{p}_1 - \frac{(\tau - t')^2}{4\tau T_1}\Delta \mathbf{p}_1 + \frac{(\tau + t')^2}{4\tau T_2}\Delta \mathbf{p}_2 \tag{4.4-56}$$

where $t' = T_1 - t$ is the time from the intersection of two segments. Similarly, the orientation equation of the tool frame is obtained as

$$\mathbf{R}(t) = \mathbf{R}_1 \,\mathrm{Rot}\left[\mathbf{n}_1, -\frac{(\tau - t')^2}{4\tau T_1}\theta_1\right] \mathrm{Rot}\left[\mathbf{n}_2, -\frac{(\tau + t')^2}{4\tau T_2}\theta_2\right] \tag{4.4-57}$$

where

$$\mathrm{Rot}\,(\mathbf{n}_1, \theta_1) = \mathbf{R}_0^{-1}\mathbf{R}_1 \qquad \text{and} \qquad \mathrm{Rot}\,(\mathbf{n}_2, \theta_2) = \mathbf{R}_1^{-1}\mathbf{R}_2$$

The last two terms represent the respective rotation matrix in quaternion form. The above equations for the position and orientation of the tool frame along the straight-line path produce a smooth transition between the two segments. It is worth pointing out that the angular acceleration will not be constant unless the axes $\mathbf{n}_1$ and $\mathbf{n}_2$ are parallel or unless one of the spin rates

$$\phi_1 = \frac{\theta_1}{T_1} \qquad \text{or} \qquad \phi_2 = \frac{\theta_2}{T_2}$$

is zero.

Bounded Deviation Joint Path. The cartesian path control scheme described above requires a considerable amount of computation time, and it is difficult to deal with the constraints on the joint-variable space behavior of the manipulator in real time. Several possible ways are available to deal with this problem. One

could precompute and store the joint solution by simulating the real time algorithm before the execution of the motion. Then motion execution would be trivial as the servo set points could be read readily from memory. Another possible way is to precompute the joint solution for every nth sample interval and then perform joint interpolation using low-degree polynominals to fit through these intermediate points to generate the servo set points. The difficulty of this method is that the number of intermediate points required to keep the manipulator hand acceptably close to the cartesian straight-line path depends on the particular motion being made. Any predetermined interval small enough to guarantee small deviations will require a wasteful amount of precomputation time and memory storage. In view of this, Taylor [1979] proposed a joint variable space motion strategy called *bounded deviation joint path*, which selects enough intermediate points during the preplanning phase to guarantee that the manipulator hand's deviation from the cartesian straight-line path on each motion segment stays within prespecified error bounds.

The scheme starts with a precomputation of all the joint solution vectors $\mathbf{q}_i$ corresponding to the knot points $\mathbf{F}_i$ on the desired cartesian straight-line path. The joint-space vectors $\mathbf{q}_i$ are then used as knot points for a joint-variable space interpolation strategy analogous to that used for the position equation of the cartesian control path. That is, for motion from the knot point $\mathbf{q}_0$ to $\mathbf{q}_1$, we have

$$\mathbf{q}(t) = \mathbf{q}_1 - \frac{T_1 - t}{T_1} \Delta\mathbf{q}_1 \tag{4.4-58}$$

and, for transition between $\mathbf{q}_0$ to $\mathbf{q}_1$ and $\mathbf{q}_1$ to $\mathbf{q}_2$, we have

$$\mathbf{q}(t') = \mathbf{q}_1 - \frac{(\tau - t')^2}{4\tau T_1}\Delta\mathbf{q}_1 + \frac{(\tau + t')^2}{4\tau T_2}\Delta\mathbf{q}_2 \tag{4.4-59}$$

where $\Delta\mathbf{q}_1 = \mathbf{q}_1 - \mathbf{q}_2$, $\Delta\mathbf{q}_2 = \mathbf{q}_2 - \mathbf{q}_1$, and T_1, T_2, τ, and t' have the same meaning as discussed before. The above equations achieve uniform velocity between the joint knot points and make smooth transitions with constant acceleration between segments. However, the tool frame may deviate substantially from the desired straight-line path. The deviation error can be seen from the difference between the $\mathbf{F}_j(t)$, which corresponds to the manipulator hand frame at the joint knot point $\mathbf{q}_j(t)$, and $\mathbf{F}_d(t)$, which corresponds to the the manipulator hand frame at the cartesian knot point $\mathbf{F}_j(t)$. Defining the displacement and rotation deviations respectively as

$$\delta_p = |\mathbf{p}_j(t) - \mathbf{p}_d(t)| \tag{4.4-60}$$

$$\delta_R = |\text{angle part of Rot}(\mathbf{n}, \phi) = \mathbf{R}_d^{-1}(t)\,\mathbf{R}_j(t)| \tag{4.4-61}$$

$$= |\phi|$$

and specifying the maximum deviations, $\delta_p^{\max}$ and $\delta_R^{\max}$, for the displacement and orientation parts, respectively, we would like to bound the deviation errors as

$$\delta_p \leqslant \delta_p^{\max} \qquad \text{and} \qquad \delta_R \leqslant \delta_R^{\max} \tag{4.4-62}$$

With this deviation error bounds, we need to select enough intermediate points between two consecutive joint knot points such that Eq. (4.4-62) is satisfied. Taylor [1979] presented a bounded deviation joint path which is basically a recursive bisector method for finding the intermediate points such that Eq. (4.4-62) is satisfied. The algorithm converges quite rapidly to produce a good set of intermediate points, though they are not a minimal set. His algorithm is as follows.

Algorithm BDJP: Given the maximum deviation error bounds $\delta_p^{\max}$ and $\delta_R^{\max}$ for the position and orientation of the tool frame, respectively, and the cartesian knot points $\mathbf{F}_i$ along the desired straight-line path, this algorithm selects enough joint knot points such that the manipulator hand frame will not deviate more than the prespecified error bounds along the desired straight-line path.

S1. *Compute joint solution.* Compute the joint solution vectors $\mathbf{q}_0$ and $\mathbf{q}_1$ corresponding to $\mathbf{F}_0$ and $\mathbf{F}_1$, respectively.

S2. *Find joint space midpoint.* Compute the joint-variable space midpoint

$$\mathbf{q}_m = \mathbf{q}_1 - \tfrac{1}{2}\Delta\mathbf{q}_1$$

where $\Delta\mathbf{q}_1 = \mathbf{q}_1 - \mathbf{q}_0$, and use $\mathbf{q}_m$ to compute the hand frame $\mathbf{F}_m$ corresponding to the joint values $\mathbf{q}_m$.

S3. *Find cartesian space midpoint.* Compute the corresponding cartesian path midpoint $\mathbf{F}_c$:

$$\mathbf{p}_c = \frac{\mathbf{p}_0 + \mathbf{p}_1}{2} \qquad \text{and} \qquad \mathbf{R}_c = \mathbf{R}_1 \,\text{Rot}\left(\mathbf{n}_1, -\frac{\theta}{2}\right)$$

where $\text{Rot}(\mathbf{n}, \theta) = \mathbf{R}_0^{-1}\mathbf{R}_1$.

S4. *Find the deviation errors.* Compute the deviation error between $\mathbf{F}_m$ and $\mathbf{F}_c$,

$$\delta_p = |\mathbf{p}_m - \mathbf{p}_c| \qquad \delta_R = |\text{angle part of Rot}(\mathbf{n}, \phi) = \mathbf{R}_c^{-1}\mathbf{R}_m|$$

$$= |\phi|$$

S5. *Check error bounds.* If $\delta_p \leqslant \delta_p^{\max}$ and $\delta_R \leqslant \delta_R^{\max}$, then stop. Otherwise, compute the joint solution vector $\mathbf{q}_c$ corresponding to the cartesian space midpoint $\mathbf{F}_c$, and apply steps S2 to S5 recursively for the two subsegments by replacing $\mathbf{F}_1$ with $\mathbf{F}_c$ and $\mathbf{F}_c$ with $\mathbf{F}_0$.

Convergence of the above algorithm is quite rapid. The maximum deviation error is usually reduced by approximately a factor of 4 for each recursive iteration.

Taylor [1979] investigated the rate of convergence of the above algorithm for a cylindrical robot (two prismatic joints coupled with a rotary joint) and found that it ranges from a factor of 2 to a factor of 4 depending on the positions of the manipulator hand.

In summary, the bounded deviation joint path scheme relies on a preplanning phase to interpolate enough intermediate points in the joint-variable space so that the manipulator may be driven in the joint-variable space without deviating more than a prespecified error from the desired straight-line path.

4.4.3 Cubic Polynomial Joint Trajectories with Torque Constraint

Taylor's straight-line trajectory planning schemes generate the joint-space vectors $\{\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t)\}$ along the desired cartesian path without taking the dynamics of the manipulator into consideration. However, the actuator of each joint is subject to saturation and cannot furnish an unlimited amount of torque and force. Thus, torque and force constraints must be considered in the planning of straight-line trajectory. This suggests that the control of the manipulator should be considered in two coherent phases of execution: off-line optimum trajectory planning, followed by on-line path tracking control.

In planning a cartesian straight-line trajectory, the path is constrainted in cartesian coordinates while the actuator torques and forces at each joint is bounded in joint coordinates. Hence, it becomes an optimization problem with mixed constraints (path and torque constraints) in two different coordinate systems. One must either convert the cartesian path into joint paths by some low-degree polynomial function approximation and optimize the joint paths and control the robot at the joint level (Lee and Chung [1984]); or convert the joint torque and force bounds into their corresponding cartesian bounds and optimize the cartesian path and control the robot at the hand level (Lee and Lee [1984]).

Although it involves numerous nonlinear transformations between the cartesian and joint coordinates, it is easier to approach the trajectory planning problem in the joint-variable space. Lin et al. [1983] proposed a set of joint spline functions to fit the segments among the selected knot points along the given cartesian path. This approach involves the conversion of the desired cartesian path into its functional representation of N joint trajectories, one for each joint. Since no transformation is known to map the straight-line path into its equivalent representation in the joint-variable space, the curve fitting methods must be used to approximate the cartesian path. Thus, to approximate a desired cartesian path in the joint-variable space, one can select enough knot points along the path, and each path segment specified by two adjacent knot points can then be interpolated by N joint polynomial functions, one function for each joint trajectory. These functions must pass through the selected knot points. Since cubic polynomial trajectories are smooth and have small overshoot of angular displacement between two adjacent knot points, Lin et al. [1983] adopted the idea of using cubic spline polynomials to fit the segment between two adjacent knots. Joint displacements for the $n - 2$

selected knot points are interpolated by piecewise cubic polynomials. In order to satisfy the continuity conditions for the joint displacement, velocity, and acceleration on the entire trajectory for the cartesian path, two extra knot points with unspecified joint displacements must be added to provide enough degrees of freedom for solving the cubic polynomials under continuity conditions. Thus, the total number of knot points becomes n and each joint trajectory consists of $n - 1$ piecewise cubic polynomials. Using the continuity conditions, the two extra knot points are then expressed as a combination of unknown variables and known constants. Then, only $n - 2$ equations need to be solved. The resultant matrix equation has a banded structure which facilitates computation. After solving the matrix equation, the resulting spline functions are expressed in terms of time intervals between adjacent knots. To minimize the total traversal time along the path, these time intervals must be adjusted subject to joint constraints within each of the $n - 1$ cubic polynomials. Thus, the problem reduces to an optimization of minimizing the total traveling time by adjusting the time intervals.

Let $\mathbf{H}(t)$ be the hand coordinate system expressed by a 4×4 homogeneous transformation matrix. The hand is required to pass through a sequence of n cartesian knot points, $[\mathbf{H}(t_1), \mathbf{H}(t_2), \ldots, \mathbf{H}(t_n)]$. The corresponding joint position vectors, $(q_{11}, q_{21}, \ldots, q_{N1})$, $(q_{12}, q_{22}, \ldots, q_{N2})$, $\ldots$, $(q_{1n}, q_{21}, \ldots, q_{Nn})$, at these n cartesian knot points can be solved using the inverse kinematics routine, where q_{ji} is the angular displacement of joint j at the ith knot point corresponding to $\mathbf{H}_i(t)$. Thus, the objective is to find a cubic polynomial trajectory for each joint j which fits the joint positions $[q_{j1}(t_1), q_{j2}(t_2), \ldots, q_{jn}(t_n)]$, where $t_1 < t_2 < \cdots < t_n$ is an ordered time sequence indicating when the hand should pass through these joint knot points. At the initial time $t = t_1$ and the final time $t = t_n$, the joint displacement, velocity, and acceleration are specified, respectively, as q_{j1}, v_{j1}, a_{j1} and q_{jn}, v_{jn}, a_{jn}. In addition, joint displacements q_{jk} at $t = t_k$ for $k = 3, 4, \ldots, n-2$ are also specified for the joint trajectory to pass through. However, q_2 and q_{n-1} are not specified; these are the two extra knot points required to provide the freedom for solving the cubic polynomials.

Let $Q_{ji}(t)$ be the piecewise cubic polynomial function for joint j between the knot points $\mathbf{H}_i$ and $\mathbf{H}_{i+1}$, defined on the time interval $[t_i, t_{i+1}]$. Then the problem is to spline $Q_{ji}(t)$, for $i = 1, 2, \ldots, n-1$, together such that the required displacement, velocity, and acceleration are satisfied and are continuous on the entire time interval $[t_1, t_n]$. Since the polynomial $Q_{ji}(t)$ is cubic, its second-time derivative $\ddot{Q}_{ji}(t)$ must be a linear function of time t,

$$Q_{ji}(t) = \frac{t_{i+1} - t}{u_i} \ddot{Q}_{ji}(t_i) + \frac{(t - t_i)}{u_i} \ddot{Q}_{ji}(t_{i+1}) \qquad \begin{matrix} i = 1, \ldots, n-1 \\ j = 1, \ldots, N \end{matrix} \tag{4.4-63}$$

where $u_i = t_{i+1} - t_i$ is the time spent in traveling segment i. Integrating $\ddot{Q}_{ji}(t)$ twice and satisfying the boundary conditions of $Q_{ji}(t_i) = q_{ji}$ and

$Q_{ji}(t_{i+1}) = q_{j,i+1}$ leads to the following interpolating functions:

$$
\begin{aligned}
Q_{ji}(t) = {} & \frac{\ddot{Q}_{ji}(t_i)}{6u_i}(t_{i+1} - t)^3 + \frac{\ddot{Q}_{ji}(t_{i+1})}{6u_i}(t - t_i)^3 \\
& + \left[\frac{q_{j,i+1}}{u_i} - \frac{u_i \ddot{Q}_{ji}(t_{i+1})}{6} \right](t - t_i) \\
& + \left[\frac{q_{j,i}}{u_i} - \frac{u_i \ddot{Q}_{ji}(t_i)}{6} \right](t_{i+1} - t) \\
& i = 1, \ldots, n-1 \\
& j = 1, \ldots, N
\end{aligned} \tag{4.4-64}
$$

Thus, for $i = 1, 2, \ldots, n-1$, $Q_{ji}(t)$ is determined if $\ddot{Q}_{ji}(t_i)$ and $\ddot{Q}_{ji}(t_{i+1})$ are known. This leads to a system of $n-2$ linear equations with unknowns $\ddot{Q}_{ji}(t_i)$ for $i = 2, \ldots, n-1$ and knowns u_i for $i = 1, 2, \ldots, n-1$,

$$
\mathbf{A}\ddot{\mathbf{Q}} = \mathbf{b} \tag{4.4-65}
$$

where

$$
\ddot{\mathbf{Q}} = \begin{bmatrix} \ddot{Q}_{j2}(t_2) \\ \ddot{Q}_{j3}(t_3) \\ \cdot \\ \cdot \\ \cdot \\ \ddot{Q}_{j,n-1}(t_{n-1}) \end{bmatrix}
$$

$$
\mathbf{A} = \begin{bmatrix}
3u_1 + 2u_2 + \dfrac{u_1^2}{u_2} & u_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
u_2 - \dfrac{u_1^2}{u_2} & 2(u_2 + u_3) & u_3 & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & u_3 & 2(u_3 + u_4) & \cdot & \cdot & \cdot & \cdot & \\
\cdot & 0 & u_4 & 2(u_4 + u_{n-3}) & \cdot & u_{n-3} & & \\
\cdot & \cdot & 0 & \cdot & \cdot & \cdot & 2(u_{n-3} + u_{n-2}) & u_{n-2} - \dfrac{u_{n-1}^2}{u_{n-2}} \\
0 & 0 & 0 & 0 & 0 & 0 & u_{n-2} & 3u_{n-1} + 2u_{n-2} + \dfrac{u_{n-1}^2}{u_{n-2}}
\end{bmatrix}
$$

and

$$\mathbf{b}=\begin{bmatrix} 6\left(\frac{q_{j3}}{u_2}+\frac{q_{j1}}{u_1}\right)-6\left(\frac{1}{u_1}+\frac{1}{u_2}\right)\left(q_{j1}+u_1v_{j1}+\frac{u_1^2}{3}a_{j1}\right)-u_1a_{j1} \\ \frac{6}{u_2}\left(q_{j1}+u_1v_{j1}+\frac{u_1^2}{3}a_{j1}\right)+\frac{6q_{j4}}{u_3}-6\left(\frac{1}{u_2}+\frac{1}{u_3}\right)q_{j3} \\ 6\left(\frac{q_{j5}-q_{j4}}{u_4}-\frac{q_4-q_{j3}}{u_3}\right) \\ 6\left(\frac{q_{j5}-q_{j4}}{u_4}-\frac{q_4-q_{j3}}{u_3}\right) \\ \vdots \\ \frac{6}{u_{n-2}}\left(q_{jn}-v_{jn}u_{n-1}+\frac{u_{n-1}^2}{3}a_{jn}\right)-6\left(\frac{1}{u_{n-2}}+\frac{1}{u_{n-3}}\right)q_{j,n-2}+\frac{6}{u_{n-3}}q_{j,n-3} \\ -6\left(\frac{1}{u_{n-1}}+\frac{1}{u_{n-2}}\right)\left(q_{jn}-v_{jn}u_{n-1}+\frac{u_{n-1}^2}{3}a_{jn}\right)+\frac{6q_{jn}}{u_{n-1}}+\frac{6q_{n-2}}{u_{n-1}u_{n-2}}-u_{n-1}a_{jn} \end{bmatrix}$$

The banded structure of the matrix $\mathbf{A}$ makes it easy to solve for $\ddot{\mathbf{Q}}$ which is substituted into Eq. (4.4-63) to obtain the resulting solution $Q_{ji}(t)$. The resulting solution $Q_{ji}(t)$ is given in terms of time intervals u_i and the given values of joint displacements, velocities, and accelerations. The above banded matrix $\mathbf{A}$ of Eq. (4.4-64) is always nonsingular if the time intervals u_i are positive. Thus, the cubic polynomial joint trajectory always has a unique solution.

Since the actuator of each joint motor is subject to saturation and cannot furnish an unlimited amount of torque and force, the total time spent on traveling the specified path approximated by the cubic polynomials is constrainted by the maximum values of each joint velocity, acceleration, and jerk which is the rate of change of acceleration. In order to maximize the speed of traversing the path, the total traveling time for the manipulator must be minimized. This can be achieved by adjusting the time intervals u_i between two adjacent knot points subject to the velocity, acceleration, jerk, and torque constraints. The problem can then be stated as:

Minimize the objective function

$$T = \sum_{i=1}^{n-1} u_i \tag{4.4-66}$$

subject to the following constraints:

Velocity constraint: $|\dot{Q}_{ji}(t)| \leqslant V_j \qquad \begin{matrix} j = 1, \ldots, N \\ i = 1, \ldots, n-1 \end{matrix}$

Acceleration constraint: $|\ddot{Q}_{ji}(t)| \leqslant A_j$ $\quad j = 1, \ldots, N$, $\quad i = 1, \ldots, n-1$

Jerk constraint: $\left| \frac{d^3}{dt^3} Q_{ji}(t) \right| \leqslant J_j$ $\quad j = 1, \ldots, N$, $\quad i = 1, \ldots, n-1$

Torque constraint: $|\tau_j(t)| \leqslant \Gamma_j$ $\quad j = 1, \ldots, N$

where T is the total traveling time, V_j, A_j, J_j, and Γ_j are, respectively, the velocity, acceleration, jerk, and torque limits of joint j.

The above constraints can be expressed in explicit forms as follows.

Velocity Constraints. Differentiating Eq. (4.4-64), and replacing $\ddot{Q}_{ji}(t_i)$ and $\ddot{Q}_{ji}(t_{i+1})$ by ω_{ji} and $\omega_{j,i+1}$, respectively, leads to the following expressions for $\dot{Q}_{ji}(t)$ and $\ddot{Q}_{ji}(t)$:

$$\dot{Q}_{ji}(t) = \frac{\omega_{ji}}{2u_i}(t_{i+1} - t)^2 + \frac{\omega_{j,i+1}}{2u_i}(t - t_i)^2 + \left(\frac{q_{j,i+1}}{u_i} - \frac{u_i \omega_{j,i+1}}{6} \right) - \left(\frac{q_{ji}}{u_i} - \frac{u_i \omega_{ji}}{6} \right) \tag{4.4-67}$$

and $$\ddot{Q}_{ji}(t) = \frac{\omega_{j,i+1}}{u_i}(t - t_i) - \frac{\omega_{ji}}{u_i}(t - t_{i+1}) \tag{4.4-68}$$

where ω_{ji} is the acceleration at $\mathbf{H}_i$ and equal to $\ddot{Q}_{ji}(t_i)$ if the time instant at which $Q_{ji}(t)$ passes through $\mathbf{H}_i$ is t_i.

The maximum absolute value of velocity exists at t_i, t_{i+1}, or $\bar{t}_i$, where $\bar{t}_i \in [t_i, t_{i+1}]$ and satisfies $\ddot{Q}_{ji}(\bar{t}_i) = 0$. The velocity constraints then become

$$\max_{t \in [t_i, t_{i+1}]} |\dot{Q}_{ji}| = \max [\, |\dot{Q}_{ji}(t_i)|, \ |\dot{Q}_{ji}(t_{i+1})|, \ |\dot{Q}_{ji}(\bar{t}_i)| \,] \leqslant V_j \tag{4.4-69}$$

$$i = 1, 2, \ldots, n-1$$

$$j = 1, 2, \ldots, N$$

where

$$|\dot{Q}_{ji}(t_i)| = \left| \frac{\omega_{ji}}{2} u_i + \frac{q_{j,i+1} - q_{ji}}{u_i} + \frac{(\omega_{ji} - \omega_{j,i+1})u_i}{6} \right|$$

$$|\dot{Q}_{ji}(t_{i+1})| = \left| \frac{\omega_{j,i+1}}{2} u_i + \frac{q_{j,i+1} - q_{ji}}{u_i} + \frac{(\omega_{ji} - \omega_{j,i+1})u_i}{6} \right|$$

and

$$|\dot{Q}_{ji}(\bar{t}_i)| = \begin{cases} \left| \dfrac{\omega_{ji}\omega_{j,i+1}u_i}{2(\omega_{ji} - \omega_{j,i+1})} + \dfrac{(\omega_{ji} - \omega_{j,i+1})u_i}{6} + \dfrac{q_{j,i+1} - q_{ji}}{u_i} \right| \\ \qquad \text{if } \omega_{ji} \neq \omega_{j,i+1} \text{ and } \bar{t}_i \in [t_i, t_{i+1}] \\ 0 \qquad \text{if } \omega_{ji} = \omega_{j,i+1} \text{ or } \bar{t}_i \notin [t_i, t_{i+1}] \end{cases}$$

Acceleration Constraints. The acceleration is a linear function of time between two adjacent knot points. Thus, the maximum absolute value of acceleration occurs at either t_i or t_{i+1} and equals the maximum of $\{|\omega_{ji}|, |\omega_{j,i+1}|\}$. Thus, the acceleration constraints become

$$\max \{|\omega_{j1}|, |\omega_{j2}|, \ldots, |\omega_{jn}|\} \leqslant A_j \qquad j = 1, 2, \ldots, N \quad (4.4\text{-}70)$$

Jerk Constraints. The jerk is the rate of change of acceleration. Thus the constraints are represented by

$$\left| \frac{\omega_{j,i+1} - \omega_{ji}}{u_i} \right| \leqslant J_j \qquad \begin{matrix} j = 1, 2, \ldots, N \\ i = 1, 2, \ldots, n-1 \end{matrix} \quad (4.4\text{-}71)$$

Torque Constraints. The torque $\tau(t)$ can be computed from the dynamic equations of motion [Eq. (3.2-25)]

$$\tau_j(t) = \sum_{k=1}^{N} D_{jk}(\mathbf{Q}_i(t))\ddot{Q}_{ji}(t) + \sum_{k=1}^{N}\sum_{m=1}^{N} h_{jkm}(\mathbf{Q}_i(t))\dot{Q}_{ki}(t)\dot{Q}_{mi}(t) + c_j(\mathbf{Q}_i(t)) \quad (4.4\text{-}72)$$

where

$$\mathbf{Q}_i(t) = (Q_{1i}(t), Q_{2i}(t), \ldots, Q_{Ni}(t))^T \qquad \begin{matrix} j = 1, 2, \ldots, N \\ i = 1, 2, \ldots, n-1 \end{matrix}$$

If the torque constraints are not satisfied, then dynamic time scaling of the trajectory must be performed to ensure the satisfaction of the torque constraints (Lin and Chang [1985], Hollerbach [1984]).

With this formulation, the objective is to find an appropriate optimization algorithm that will minimize the total traveling time subject to the velocity, acceleration, jerk, and torque constraints. There are several optimization algorithms available, and Lin et al. [1983] utilized Nelder and Mead's flexible polyhedron search to obtain an iterative algorithm which minimizes the total traveling time subject to the constraints on joint velocities, accelerations, jerks, and torques. Results using this optimization technique can be found in Lin et al. [1983].

4.5 CONCLUDING REMARKS

Two major approaches for trajectory planning have been discussed: the joint-interpolated approach and the cartesian space approach. The joint-interpolated approach plans polynomial sequences that yield smooth joint trajectory. In order to yield faster computation and less extraneous motion, lower-degree polynomial sequences are preferred. The joint trajectory is split into several trajectory segments and each trajectory segment is splined by a low-degree polynomial. In particular, 4-3-4 and five-cubic polynomial sequences have been discussed.

Several methods have been discussed in the cartesian space planning. Because servoing is done in the joint-variable space while a path is specified in cartesian coordinates, the most common approach is to plan the straight-line path in the joint-variable space using low-degree polynomials to approximate the path. Paul [1979] used a translation and two rotations to accomplish the straight-line motion of the manipulator hand. Taylor [1979] improved the technique by using a quaternion approach to represent the rotational operation. He also developed a bounded deviation joint control scheme which involved selecting more intermediate interpolation points when the joint polynomial approximation deviated too much from the desired straight-line path. Lin et al. [1983] used cubic joint polynomials to spline n interpolation points selected by the user on the desired straight-line path. Then, the total traveling time along the knot points was minimized subject to joint velocity, acceleration, jerk, and torque constraints. These techniques represent a shift away from the real-time planning objective to an off-line planning phase. In essence, this decomposes the control of robot manipulators into off-line motion planning followed by on-line tracking control, a topic that is discussed in detail in Chap. 5.

REFERENCES

Further reading on joint-interpolated trajectories can be found in Paul [1972], Lewis [1973, 1974], Brady et al. [1982], and Lee et al. [1986]. Most of these joint-interpolated trajectories seldom include the physical manipulator dynamics and actuator torque limit into the planning schemes. They focused on the requirement that the joint trajectories must be smooth and continuous by specifying velocity and acceleration bounds along the trajectory. In addition to the continuity constraints, Hollerbach [1984] developed a time-scaling scheme to determine whether a planned trajectory is realizable within the dynamics and torque limits which depend on instantaneous joint position and velocity.

The design of a manipulator path made up of straight line segments in the cartesian space is discussed by Paul [1979], using the homogeneous transformation matrix to represent target positions for the manipulator hand to traverse. Movement between two consecutive target positions is accomplished by two sequential operations: a translation and a rotation to align the approach vector of the manipulator hand and a final rotation about the tool axis to align the gripper orientation.

A quadratic polynomial interpolation routine in the joint-variable space is then used to guarantee smooth transition between two connected path segments. Taylor [1979], using the quaternion representation, extended Paul's method for a better and uniform motion. In order to achieve real-time trajectory planning objective, both approaches neglect the physical manipulator torque constraint.

Other existing cartesian planning schemes are designed to satisfy the continuity and the torque constraints simultaneously. To include the torque constraint in the trajectory planning stage, one usually assumes that the maximum allowable torque is constant at every position and velocity. For example, instead of using varying torque constraint, Lin et al. [1983] and Luh and Lin [1984] used the velocity, acceleration, and jerk bounds which are assumed constant for each joint. They selected several knot points on the desired cartesian path, solved the inverse kinematics, and found appropriate smooth, lower-degree polynomial functions which guaranteed the continuity conditions to fit through these knot points in the joint-variable space. Then, by relaxing the normalized time to the servo time, the dynamic constraint with the constant torque bound assumption was included along the trajectory. Due to the joint-interpolated functions, the location of the manipulator hand at each servo instant may not be exactly on the desired path, but rather on the joint-interpolated polynomial functions.

Lee [1985] developed a discrete time trajectory planning scheme to determine the trajectory set points exactly on a given straight-line path which satisfies both the smoothness and torque constraints. The trajectory planning problem is formulated as a maximization of the distance between two consecutive cartesian set points on a given straight-line path subject to the smoothness and torque constraints. Due to the discrete time approximations of joint velocity, acceleration, and jerk, the optimization solution involves intensive computations which prevent useful applications. Thus, to reduce the computational cost, the optimization is realized by iterative search algorithms.

PROBLEMS

4.1 A single-link rotary robot is required to move from $\theta(0) = 30^\circ$ to $\theta(2) = 100^\circ$ in 2 s. The joint velocity and acceleration are both zero at the initial and final positions. (*a*) What is the highest degree polynomial that can be used to accomplish the motion? (*b*) What is the lowest degree polynomial that can be used to accomplish the motion?

4.2 With reference to Prob. 4.1, (*a*) determine the coefficients of a cubic polynomial that accomplishes the motion; (*b*) determine the coefficients of a quartic polynomial that accomplishes the motion; and (*c*) determine the coefficients of a quintic polynomial that accomplishes the motion. You may split the joint trajectory into several trajectory segments.

4.3 Consider the two-link robot arm discussed in Sec. 3.2.6, and assume that each link is 1 m long. The robot arm is required to move from an initial position $(x_0, y_0) = (1.96, 0.50)$ to a final position $(x_f, y_f) = (1.00, 0.75)$. The initial and final velocity and acceleration are zero. Determine the coefficients of a cubic polynomial for each joint to accomplish the motion. You may split the joint trajectory into several trajectory segments.

4.4 In planning a 4-3-4 trajectory one needs to solve a matrix equation, as in Eq. (4.3-46). Does the matrix inversion of Eq. (4.3-46) always exist? Justify your answer.

4.5 Given a PUMA 560 series robot arm whose joint coordinate frames have been established as in Fig. 2.11, you are asked to design a 4-3-4 trajectory for the following conditions: The initial position of the robot arm is expressed by the homogeneous transformation matrix $\mathbf{T}_{\text{initial}}$:

$$\mathbf{T}_{\text{initial}} = \begin{bmatrix} -0.660 & -0.436 & -0.612 & -184.099 \\ -0.750 & 0.433 & 0.500 & 892.250 \\ 0.047 & 0.789 & -0.612 & -34.599 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The final position of the robot arm is expressed by the homogeneous transformation matrix $\mathbf{T}_{\text{final}}$

$$\mathbf{T}_{\text{final}} = \begin{bmatrix} -0.933 & -0.064 & 0.355 & 412.876 \\ -0.122 & 0.982 & -0.145 & 596.051 \\ -0.339 & -0.179 & -0.924 & -545.869 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The lift-off and set-down positions of the robot arm are obtained from a rule of thumb by taking 25 percent of d_6 (the value of d_6 is 56.25 mm). What are the homogeneous transformation matrices at the lift-off and set-down positions (that is, $\mathbf{T}_{\text{lift-off}}$ and $\mathbf{T}_{\text{set-down}}$)?

4.6 Given a PUMA 560 series robot arm whose joint coordinate frames have been established as in Fig. 2.11, you are asked to design a 4-3-4 trajectory for the following conditions: The initial position of the robot arm is expressed by the homogeneous transformation matrix $\mathbf{T}_{\text{initial}}$:

$$\mathbf{T}_{\text{initial}} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 600.0 \\ 0 & 0 & -1 & -100.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The set-down position of the robot arm is expressed by the homogeneous transformation matrix $\mathbf{T}_{\text{set-down}}$:

$$\mathbf{T}_{\text{set-down}} = \begin{bmatrix} 0 & 1 & 0 & 100.0 \\ 1 & 0 & 0 & 400.0 \\ 0 & 0 & -1 & -50.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(*a*) The lift-off and set-down positions of the robot arm are obtained from a rule of thumb by taking 25 percent of d_6 (the value of d_6 is 56.25 mm) plus any required rotations. What is the homogeneous transformation matrix at the lift-off (that is, $\mathbf{T}_{\text{lift-off}}$) if the hand is rotated 60° about the s axis at the initial point to arrive at the lift-off point? (*b*) What is the homogeneous transformation matrix at the final position (that is, $\mathbf{T}_{\text{final}}$) if the hand is rotated −60° about the s axis at the set-down point to arrive at the final position?

4.7 A manipulator is required to move along a straight line from point **A** to point **B**, where **A** and **B** are respectively described by

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 15 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 & -1 & 0 & 20 \\ 0 & 0 & 1 & 30 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The motion from **A** to **B** consists of a translation and two rotations, as described in Sec. 4.4.1. Determine θ, ψ, ϕ and x, y, z for the drive transform. Aslo find three intermediate transforms between **A** and **B**.

4.8 A manipulator is required to move along a straight line from point **A** to point **B** rotating at constant angular velocity about a vector **k** and at an angle θ. The points **A** and **B** are given by a 4×4 homogeneous transformation matrices as

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & 0 & 10 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & -1 & 0 & 10 \\ 0 & 0 & 1 & 30 \\ -1 & 0 & 0 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Find the vector **k** and the angle θ. Also find three intermediate transforms between **A** and **B**.

4.9 Express the rotation results of Prob. 4.8 in quaternion form.

4.10 Give a quaternion representation for the following rotations: a rotation of 60° about **j** followed by a rotation of 120° about **i**. Find the resultant rotation in quaternion representation.

4.11 Show that the inverse of the banded structure matrix **A** in Eq. (4.4-64) always exists.