Chapter 6

Trajectory Planning

A trajectory is the path followed by the manipulator, plus the time profile along the path. Trajectories can be planned either in joint space (directly specifying the time evolution of the joint angles) or in Cartesian space (specifying the position and orientation of the end frame). Issues in trajectory planning include attaining a specific target from an initial starting point, avoiding obstacles, and staying within manipulator capabilities.

Planning in joint space is the simplest and fastest, because inverse kinematics are avoided. The drawback is that the end effector pose is not directly controlled, and hence collision avoidance is difficult. Planning in Cartesian space more directly allows the geometric constraints of the external world to be met, but then inverse kinematics must be solved.

6.1 Polynomial Trajectories

Let $x(t)$ represent the planning variable, whether a joint variable or a task coordinate. The simplest and most common way of specifying a trajectory segment is as a polynomial.

$$x(t) = a_0 + \sum_{i=1}^{n} a_i t^i \qquad\qquad t_o \leq t \leq t_1 \tag{6.1}$$

This n-th order polynomial is valid in the time range $t_0 \leq t \leq t_1$. There are $n+1$ coefficients a_i, which are adjusted based on constraint relations such as the beginning and end positions, velocities, or even accelerations. A given polynomial may be followed by another polynomial at time $t = t_1$, and matching constraints will arise such as continuity in position and velocity or possibly acceleration. The number of constraints has to total to $n+1$ in order to solve for the coefficients.

Generally speaking, polynomials higher than degree 5 are not used because they have too many wiggles. Instead, lower-order polynomials are spliced together to cover a larger period of time or satisfy additional constraints.

Now we consider polynomials up to degree 3, and discuss how the coefficients can be determined from boundary constraints. For the time range $t_0 \leq t \leq t_1$, let's start by supposing that $t_0 = 0$ and t_1 is specified.

6.1.1 Stationary trajectory

The robot may not be moving during a given time period. The coefficient a_0 represents whatever position the robot is sitting at.

$$x(t) = a_0 \tag{6.2}$$

6.1.2 Linear Trajectories

The simplest moving trajectory is a linear polynomial, which means moving at constant velocity a_1.

$$x(t) = a_0 + a_1 t \tag{6.3}$$

$$\dot{x}(t) = a_1 \tag{6.4}$$

To determine coefficients a_0 and a_1, one can specify the beginning position x_0 and the constant velocity x'.

$x(0)$	$\dot{x}(0)$
x_0	x'

Table 6.1: Constraints for a linear trajectory.

Straightforwardly, $a_0 = x_0$ and $a_1 = x'$.

6.1.3 Quadratic Trajectories

A quadratic trajectory means moving with a constant acceleration $2a_2$. The velocity is a linear polynomial.

$$x(t) = a_0 + a_1 t + a_2 t^2 \tag{6.5}$$

$$\dot{x}(t) = a_1 + 2a_2 t \tag{6.6}$$

$$\ddot{x}(t) = 2a_2 \tag{6.7}$$

A quadratic trajectory has coefficients a_0, a_1 and a_2 to determine, which can be done by specifying the endpoint positions x_0, x_1 and the initial velocity x'_0.

$x(0)$	$\dot{x}(0)$	$x(t_1)$
x_0	x'_0	x_1

Table 6.2: Constraints for a quadratic trajectory.

At time $t = 0$, we find that $a_0 = x_0$ from the position equation and $a_1 = x'_0$ from the velocity equation. At time $t = t_1$, the position equation yields:

$$x_1 = x_0 + x'_0 t_1 + a_2 t_1^2$$

$$a_2 = \frac{x_1 - x_0 - x'_0 t_1}{t_1^2}$$

6.1.4 Cubic Trajectories

A cubic trajectory means moving with a constant 3rd derivative $6a_3$ (jerk). The velocity is a quadratic and the acceleration is a linear polynomial.

$$\begin{aligned} x(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 && (6.8)\\ \dot{x}(t) &= a_1 + 2a_2 t + 3a_3 t^2 && (6.9)\\ \ddot{x}(t) &= 2a_2 + 6a_3 t && (6.10)\\ \frac{d^3 x(t)}{dt} &= 6a_3 && (6.11) \end{aligned}$$

A cubic trajectory has coefficients a_0, a_1, a_2 and a_3 to determine, which can be done by specifying the endpoint positions x_0, x_1 and endpoint velocities x'_0, x'_1.

$x(0)$	$\dot{x}(0)$	$x(t_1)$	$\dot{x}(t_1)$
x_0	x'_0	x_1	x'_1

Table 6.3: Constraints for a cubic trajectory.

At time $t = 0$, we find that $a_0 = x_0$ from the position equation and $a_1 = x'_0$ from the velocity equation. At time $t = t_1$, the position and velocity equations yield:

$$\begin{aligned} x_1 &= x_0 + x'_0 t_1 + a_2 t_1^2 + a_3 t_1^3 && (6.12)\\ x'_1 &= x'_0 + 2a_2 t_1 + 3a_3 t_1^2 && (6.13) \end{aligned}$$

Eliminate a_3 from (6.12) and (6.13):

$$\begin{aligned} 3x_1 - t_1 x'_1 &= 3x_0 + 3x'_0 t_1 + 3a_2 t_1^2 + 3a_3 t_1^3 - x'_0 t_1 - 2a_2 t_1^2 - 3a_3 t_1^3 \\ &= 3x_0 + 2x'_0 t_1 + a_2 t_1^2 && (6.14) \end{aligned}$$

Hence

$$a_2 = \frac{3(x_1 - x_0) - (2x'_0 + x'_1)t_1}{t_1^2} \quad (6.15)$$

Similarly,

$$a_3 = \frac{2(x_0 - x_1) + (x'_0 + x'_1)t_1}{t_1^3} \quad (6.16)$$

Example 6.1: Suppose $x_0 = 10$ degrees, $x_1 = -20$ degrees, and $x'_0 = x'_1 = 0$ degrees/second. Suppose $t_1 = 1$ second. Then

$$a_0 = 10, \quad a_1 = 0, \quad a_2 = -90, \quad , a_3 = 60$$

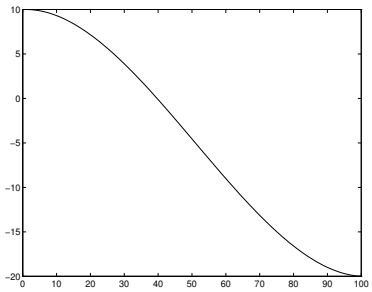

Figure 6.1: Cubic polynomial trajectory.

Figure 6.1 plots the resulting trajectory. The Appendix contains the Matlab program used to generate this figure.

Quadratic and quintic polynomials are also used. Unconstrained human arm movements are well modelled by quintic polynomials.

6.2 Multi-Segment Polynomials

6.2.1 Splicing Two Polynomial Trajectories

Suppose polynomial $x_1(t)$ is valid in the time range $0 \leq t \leq t_1$, and polynomial $x_2(t)$ is valid in the time range $t_1 \leq t \leq t_2$. They meet at time $t = t_1$. In order to facilitate splicing the trajectories together, the coefficients of the second polynomial are put into time shift form.

$$x_1(t) = a_0 + \sum_{i=1}^{n} a_i t^i \qquad 0 \leq t \leq t_1 \tag{6.17}$$

$$x_2(t) = b_0 + \sum_{i=1}^{n} b_i (t - t_1)^i \qquad t_1 \leq t \leq t_2 \tag{6.18}$$

This makes the coefficients of the second polynomial independent of whatever the starting time t_1 is. There are two kinds of constraints: unary constraints on a single polynomial, and binary (or matching) constraints on the two polynomials together.

- Unary constraints on the polynomials include starting or ending positions, velocities or accelerations.
- Binary constraints include position and velocity continuity: $x_1(t_1) = x_2(t_1)$ and $\dot{x}_1(t_1) = \dot{x}_2(t_1)$.

As a specific example, suppose polynomial $x_1(t)$ has an initial position and velocity of zero. Polynomial $x_2(t)$ has a final position of x_2 and a final velocity of zero. The two polynomials meet at position x_1 and have the same velocity there. Table 6.4 summarizes these constraints.

There are 6 unary constraints, and one binary constraint. Although the value $x_1(t_1) = x_1$ matches the value $x_2(t_1) = x_1$, they are both unary constraints because they take on a specific value x_1. The constraint $\dot{x}_1(t_1) = \dot{x}_2(t_1)$ is a binary constraint because its value is not specified. Given 7 constraints, that means one polynomial must be of degree 3 and the other of degree 2. In this case, the number of constraints on

$x_1(0)$	$\dot{x}_1(0)$	$\ddot{x}_1(0)$	$x_1(t_1)$	$\dot{x}_1(t_1)$	$\ddot{x}_1(t_1)$	$x_2(t_1)$	$\dot{x}_2(t_1)$	$\ddot{x}_2(t_1)$	$x_2(t_2)$	$\dot{x}_2(t_2)$	$\ddot{x}_2(t_1)$
0	0	–	x_1	$\dot{x}_2(t_1)$	–	x_1	–	–	x_2	0	–

Table 6.4: Constraints for splicing together the two polynomials.

each polynomial are the same, so it doesn't matter which polynomial is quadratic and which is cubic. It is simpler to make $x_2(t)$ be the degree 2 polynomial because of the time starting at zero.

$$x_1(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{6.19}$$

$$x_2(t) = b_0 + b_1(t - t_1) + b_2(t - t_1)^2 \tag{6.20}$$

- Unary constraints on $x_1(t)$. From (6.8) and (6.9),

$$x_1(0) = a_0 = 0 \tag{6.21}$$

$$\dot{x}_1(0) = a_1 = 0 \tag{6.22}$$

$$x_1(t_1) = a_2 t_1^2 + a_3 t_1^3 = x_1 \tag{6.23}$$

- Unary constraints on $x_2(t)$. From (6.5) and (6.6),

$$x_2(t_1) = b_0 = x_1 \tag{6.24}$$

$$x_2(t_2) = x_1 + b_1(t_2 - t_1) + b_2(t_2 - t_1)^2 = x_2 \tag{6.25}$$

$$\dot{x}_2(t_2) = b_1 + 2b_2(t_2 - t_1) = 0 \tag{6.26}$$

- Binary constraint $\dot{x}_1(t_1) = \dot{x}_2(t_1)$. This yields

$$2a_2 t_1 + 3a_3 t_1^2 = b_1 \tag{6.27}$$

Find b_2 by multiplying (6.26) by $(t_2 - t_1)$ and subtracting (6.25):

$$\begin{aligned} b_2(t_2 - t_1)^2 &= x_1 - x_2 \\ b_2 &= \frac{x_1 - x_2}{(t_2 - t_1)^2} \end{aligned} \tag{6.28}$$

Now solve for b_1 from (6.26).

$$\begin{aligned} b_1 &= -2b_2(t_2 - t_1) \\ &= 2\frac{x_2 - x_1}{t_2 - t_1} \end{aligned} \tag{6.29}$$

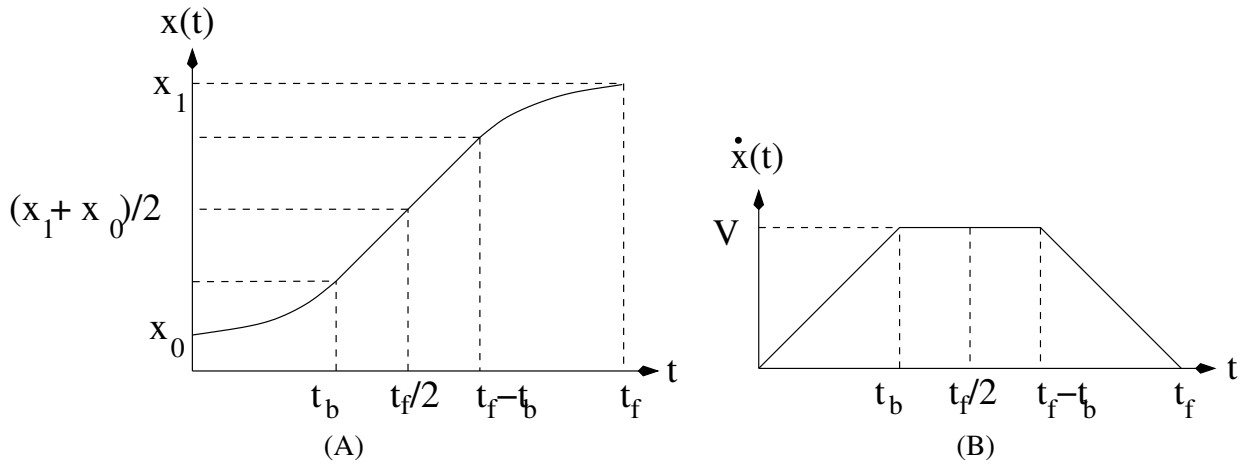

Figure 6.2: Velocity (A) and position (B) profiles of LSPB trajectory.

Solve for a_3 by multiplying (6.27) by t_1 and subtracting (6.23):

$$\begin{aligned} a_3 t_1^3 &= b_1 t_1 - 2x_1 \\ a_3 &= \frac{b_1 t_1 - 2x_1}{t_1^3} \end{aligned} \tag{6.30}$$

Finally, from (6.27),

$$a_2 = \frac{x_1 - a_3 t_1^3}{t_1^2} \tag{6.31}$$

6.2.2 Linear Segment with Parabolic Blend

A common trajectory for industrial robots is the linear segment with parabolic blend (LSPB). Joint velocities are often slew-rate limited: a maximum velocity V is set by the joint controller. To move from an initial point x_0 to a final point x_1, the joint is accelerated from zero to the maximum velocity V using a parabolic trajectory, then the joint cruises along at velocity V until it nears the endpoint, then decelerates to a stop with another quadratic polynomial (Figure 6.2). The second quadratic is symmetric to the first quadratic. Define the three polynomials as:

$$\begin{aligned} x_1(t) &= a_0 + a_1 t + a_2 t^2 & 0 &\le t \le t_b \\ x_2(t) &= b_0 + b_1 (t - t_b) & t_b &\le t \le t_f - t_b \\ x_3(t) &= c_0 + c_1 (t - (t_f - t_b)) + c_2 (t - (t_f - t_b))^2 & t_f - t_b &\le t \le t_f \end{aligned} \tag{6.32}$$

where t_b is the time of transition from the first quadratic to the linear segment, t_f is the final time, and $t_f - t_b$ is the time of transition from the linear segment to the second quadratic (due to symmetry). The time shift form of the second and third segments is employed to simplify the development. The transition time t_b is

not specified and must be solved for. Along with t_b, there are 8 polynomial coefficients to solve for, and consequently 9 constraints must be given. One way to specify 9 constraints is as follows:

Unilateral position constraints: $x_1(0) = x_0$ and $x_3(t_f) = x_1$, specified by the user.

Unilateral velocity constraints: $\dot{x}(0) = 0$, $\dot{x}_1(t_b) = V$, $\dot{x}_2(t_b) = V$, $\dot{x}_3(t_f - t_b) = V$, and $\dot{x}_3(t_f) = 0$.

Position matching constraints: $x_1(t_b) = x_2(t_b)$ and $x_2(t_f - t_b) = x_3(t_f - t_b)$.

However, the LSPB specification is by convention slightly different, substituting a symmetry constraint $x(t_f/2) = (x_0 + x_1)/2$, which expresses that at the time midpoint the position is the average of the two endpoints. This substitutes for the position matching constraint $x_2(t_f - t_b) = x_3(t_f - t_b)$, which is implied.

As mentioned above, the transition time t_b is a variable, which will be solved for given these constraints. It cannot also be specified because there would be too many constraints, leading to an inconsistency and no solution. If one wishes to adopt t_b as a constraint, then one of the other constraints would have to be dropped, such as the symmetry constraint.

1. Ramping Up

There are three unilateral constraints on the first quadratic polynomial: position and velocity at the start, and the slew velocity at its end. The velocity equation is:

$$\dot{x}_1(t) = a_1 + 2a_2 t \tag{6.33}$$

Substituting the 3 constraints into the position and velocity equations,

$$\begin{aligned} x_1(0) &= x_0 = a_0 \\ \dot{x}_1(0) &= 0 = a_1 \\ \dot{x}_1(t_b) &= V = 2a_2 t_b \end{aligned}$$

Coefficients a_0 and a_1 are found immediately, and a_2 can be expressed in terms of t_b as $a_2 = V/(2t_b)$. Substituting, the first quadratic is now:

$$x(t) = x_0 + \frac{V}{2t_b} t^2 \qquad\qquad 0 \le t \le t_b \tag{6.34}$$

2. Cruising Along

There are two constraints acting on the linear segment: the symmetry constraint and the slew velocity.

$$\begin{aligned} x_2(\frac{t_f}{2}) &= \frac{x_0 + x_1}{2} = b_0 + b_1(\frac{t_f}{2} - t_b) \\ \dot{x}_2(t) &= V = b_1 \end{aligned}$$

b_1 is found immediately, and b_0 can be expressed as:

$$\begin{aligned} \frac{x_0 + x_1}{2} &= b_0 + V(\frac{t_f}{2} - t_b) \\ b_0 &= \frac{x_0 + x_1}{2} - V(\frac{t_f}{2} - t_b) \end{aligned}$$

Substituting, the linear segment is:

$$x_2(t) = \frac{x_0 + x_1}{2} - V(\frac{t_f}{2} - t_b) + V(t - t_b) \qquad t_b \le t \le t_f - t_b \tag{6.35}$$

The positions of the first two polynomials at time t_b must match. Equating (6.34) to (6.35) at $t = t_b$,

$$\begin{aligned} x_0 + \frac{V}{2t_b}t_b^2 &= \frac{x_0 + x_1}{2} - V(\frac{t_f}{2} - t_b) \\ t_b &= t_f + \frac{x_0 - x_1}{V} \end{aligned} \tag{6.36}$$

3. Ramping Down

The three unilateral constraints on the second quadratic are that $x_3(t_f) = x_1$, $\dot{x}_3(t_f) = 0$, and $\dot{x}_3(t_f - t_b) = V$. The velocity equation is:

$$\dot{x}_3(t) = c_1 + 2c_2(t - (t_f - t_b)) \tag{6.37}$$

Substituting the three constraints into the position and velocity equations,

$$\begin{aligned} x_3(t_f) &= c_0 + c_1 t_b + c_2 t_b^2 = x_1 \\ \dot{x}_3(t_f) &= c_1 + 2c_2 t_b = 0 \\ \dot{x}_3(t_f - t_b) &= c_1 = V \end{aligned}$$

The other two coefficients can now be found:

$$\begin{aligned} c_2 &= -\frac{V}{2t_b} \\ c_0 &= x_1 - \frac{V t_b}{2} \end{aligned}$$

Example 6.2: Suppose $x_0 = 0$, $x_1 = 40$, $V = 60$, and $t_f = 1$. Then $t_b = 1/3$. Figure 6.3 plots the resulting trajectory.

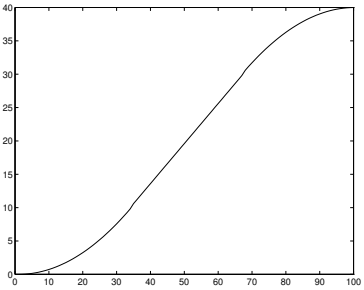

Figure 6.3: Linear segment with parabolic blends.

6.2.3 Cartesian Trajectory Planning

Cartesian trajectories can be fashioned just like joint trajectories. In the present section, we expand the types of trajectories to include *via points.* A via point is an intermediate point that the trajectory passes close to, but does not necessarily pass through. An analogy might be knot points in b-splines in computer graphics. We suppose that a Cartesian trajectory is composed of a series of straight-line, constant-velocity trajectories connected by via points. Both position and orientation of the end-link frame have to be interpolated from beginning to end of each trajectory segment.

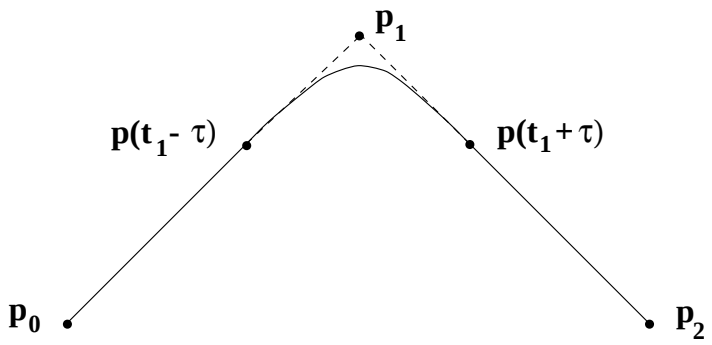

Figure 6.4: Cartesian straight-line segments and transition controlled by via point $\mathbf{p}_1$.

The key issue is the shape of the curved trajectory that links two straight-line segments (Figure 6.4). We present a well-known solution due to Russ Taylor [2]. We will first consider how to plan the position trajectory $\mathbf{p}(t)$.

- The first straight-line trajectory segment starts at point $\mathbf{p}_0$. Without any transition, the trajectory would arrive at via point $\mathbf{p}_1$ in time t_1 under constant velocity $\Delta\mathbf{p}_1/t_1$, where $\Delta\mathbf{p}_1 = \mathbf{p}_1 - \mathbf{p}_0$.
- The second straight-line trajectory ends at point $\mathbf{p}_2$. Without any transition, the trajectory would begin at via point $\mathbf{p}_1$ and arrive at $\mathbf{p}_2$ in time t_2 under constant velocity $\Delta\mathbf{p}_2/t_2$, where $\Delta p_2 = \mathbf{p}_2 - \mathbf{p}_1$.
- At a time interval τ before the scheduled arrival at $\mathbf{p}_1$, we start the curved transition. Thus $\mathbf{p}(t_1 - \tau)$ is the first transition point. Similarly, $\mathbf{p}(t_1 + \tau)$ is the second transition point, from curved to the second straight-line trajectory. The curved transition is a parabolic segment, i.e., the acceleration is constant.

Transition times t_1 and t_2, and points $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$ are prespecified. The polynomial $\mathbf{p}(t)$ as a function of t is:

$$\mathbf{p}(t) = \begin{cases} \mathbf{p}_1 - \dfrac{t_1 - t}{t_1}\Delta\mathbf{p}_1 & 0 \le t \le t_1 - \tau \\ \mathbf{a}_0 + \mathbf{a}_1(t - (t_1 - \tau)) + \mathbf{a}_2(t - (t_1 - \tau))^2 & t_1 - \tau \le t \le t_1 + \tau \\ \mathbf{p}_1 + \dfrac{t - t_1}{t_2}\Delta\mathbf{p}_2 & t_1 + \tau \le t \le t_1 + t_2 \end{cases}$$

By convention, the Taylor trajectory is specified in terms of the via point $\mathbf{p}_1$. Thus for the first segment, the position is measured backwards from $\mathbf{p}_1$. This allows real-time trajectory modification by moving $\mathbf{p}_1$ if the obstacle moves.

There are matching constraints in position and velocity at $t = t_1 - \tau$ and in velocity at $t = t_1 + \tau$. The matching constraint in position at $t = t_1 + \tau$ is redundant.

$$\begin{aligned} \mathbf{p}_1 - \frac{\tau}{t_1}\Delta\mathbf{p}_1 &= \mathbf{a}_0 \\ \frac{\Delta\mathbf{p}_1}{t_1} &= \mathbf{a}_1 \\ \frac{\Delta\mathbf{p}_2}{t_2} &= \mathbf{a}_1 + 4\mathbf{a}_2\tau \end{aligned}$$

Subtract the two velocity constraints to find $\mathbf{a}_2$:

$$\mathbf{a}_2 = \frac{1}{4\tau}\left(\frac{\Delta\mathbf{p}_2}{t_2} - \frac{\Delta\mathbf{p}_1}{t_1}\right) \tag{6.38}$$

The quadratic polynomial then becomes:

$$\begin{aligned} \mathbf{p}(t) &= \mathbf{p}_1 - \frac{\tau}{t_1}\Delta\mathbf{p}_1 + \frac{\Delta\mathbf{p}_1}{t_1}(t - (t_1 - \tau)) + \frac{1}{4\tau}\left(\frac{\Delta\mathbf{p}_2}{t_2} - \frac{\Delta\mathbf{p}_1}{t_1}\right)(t - (t_1 - \tau))^2 \\ &= \mathbf{p}_1 - \frac{\Delta\mathbf{p}_1}{4\tau t_1}(t - t_1 - \tau)^2 + \frac{\Delta\mathbf{p}_2}{4\tau t_2}(t - t_1 + \tau)^2 \end{aligned} \tag{6.39}$$

after some simplification.

6.3 Interpolating 3D Rotations

It is also necessary to interpolation orientation smoothly along a trajectory. Applying the angle-axis formula is the best way to do this. Suppose coordinate system 0 is the reference coordinate system, and ${}^0\mathbf{R}_1$ is the orientation at the beginning of a trajectory segment and ${}^0\mathbf{R}_2$ is the orientation at the end of the segment. The difference in orientation from start to end is then ${}^1\mathbf{R}_2$, which can be found by inverting ${}^0\mathbf{R}_1$.

$$\begin{aligned} {}^0\mathbf{R}_1\,{}^1\mathbf{R}_2 &= {}^0\mathbf{R}_2 \\ {}^1\mathbf{R}_2 &= {}^0\mathbf{R}_1^T\,{}^0\mathbf{R}_2 \end{aligned} \tag{6.40}$$

In turn, the difference in orientation can be expressed in terms of the angle-axis formula using the method of Chapter 3: ${}^1\mathbf{R}_2 = \mathbf{R}_k(\theta_{12})$ for some $\mathbf{k}$ and θ_{12}. By creating a polynomial trajectory $\theta(t)$, orientation can be smoothly interpolated. For example, a linear trajectory would be:

$$\theta(t) \quad = \quad \theta_{12}\frac{t}{t_{12}} \qquad 0 \leq t \leq t_{12} \tag{6.41}$$

where t_{12} is the duration of the trajectory segment. Then the time-varying orientation trajectory $\mathbf{R}(t)$ is

$$\mathbf{R}(t) = {}^0\mathbf{R}_1 \; \mathbf{R}_k(\theta(t)) \tag{6.42}$$

At the beginning, $\theta(0) = 0$ and $\mathbf{R}(0) = {}^0\mathbf{R}_1 \; \mathbf{R}_k(0) = {}^0\mathbf{R}_1$. At the end, $\theta(t_{12}) = \theta_{12}$ and $\mathbf{R}(t_{12}) = {}^0\mathbf{R}_1 \; \mathbf{R}_k(\theta(t_{12})) = {}^0\mathbf{R}_2$. In between, orientation is linearly interpolated about the fixed axis $\mathbf{k}$.

Higher-order polynomials for $\theta(t)$ can be used. For example, the Taylor trajectories use a quadratic polynomial.

Another Cartesian trajectory planning scheme is due to Paul [1].

$$\mathbf{R}(t) \quad = \quad \mathbf{R}_0\mathbf{R}_k(\theta t/t_{12})\mathbf{R}_z(\phi t/t_{12}) \tag{6.43}$$

where $\mathbf{z}$ is the approach axis of the tool frame (see Chapter 4), ϕ is the twist, and $\mathbf{k}$ is the cross product of the initial and final $\mathbf{z}$ axes. An advantage of Paul's scheme is the planning of rotation in terms of tool axes. An advantage of the angle-axis rotational scheme is uniformity of motion.

6.3.1 Taylor Trajectory for Orientation

In terms of the Taylor trajectory, let $\mathbf{R}_0$ be the orientation at the start, $\mathbf{R}_1$ the orientation at the via point, and $\mathbf{R}_2$ the orientation at the goal. Then

$$\begin{aligned} \mathbf{R}_0\mathbf{R}_{k_1}(\theta_1) &= \mathbf{R}_1 \\ \mathbf{R}_{k_1}(\theta_1) &= \mathbf{R}_0^T\mathbf{R}_1 \end{aligned} \tag{6.44}$$

$$\begin{aligned} \mathbf{R}_1\mathbf{R}_{k_2}(\theta_2) &= \mathbf{R}_2 \\ \mathbf{R}_{k_2}(\theta_2) &= \mathbf{R}_1^T\mathbf{R}_2 \end{aligned} \tag{6.45}$$

where the angle-axis representation $\mathbf{R}_{k_1}(\theta_1)$ transforms from $\mathbf{R}_1$ to $\mathbf{R}_0$, and $\mathbf{R}_{k_2}(\theta_2)$ transforms from $\mathbf{R}_2$ to $\mathbf{R}_1$.

The amount of rotation about $\mathbf{k}_1$ and about $\mathbf{k}_2$ can then be made a linear function of time. The rotational motion along the first straight-line path is:

$$\mathbf{R}(t) = \mathbf{R}_1\mathbf{R}_{k_1}(-\frac{t_1 - t}{t_1}\theta_1), \qquad 0 \leq t \leq t_1 - \tau \tag{6.46}$$

The rotational motion along the second straight-line path is:

$$\mathbf{R}(t) = \mathbf{R}_1\mathbf{R}_{k_2}(\frac{t - t_1}{t_2}\theta_2), \qquad t_1 + \tau \leq t \leq t_1 + t_2 \tag{6.47}$$

The transitions between path segments will now be reexamined to incorporate rotations. The results are stated, and may be derived analogously to the position transition equations (6.39):

$$\mathbf{R}(t) = \mathbf{R}_1\mathbf{R}_{k_1}(-\theta_1\frac{(t-t_1-\tau)^2}{4\tau t_1})\mathbf{R}_{k_2}(\theta_2\frac{(t-t_1+\tau)^2}{4\tau t_2}), \qquad t_1-\tau \le t \le t_1+\tau \tag{6.48}$$

Taylor represents rotations as quaternions for efficiency.

6.4 Appendix

```
% cubic  Cubic polynomial trajectory with specified end p & v, and tf.
%
%       q = cubic(q0,q1,qdot0,qdot1,tf) generates a cubic polynomial
%       trajectory comprised of 100 equally-spaced points in time.
%
%       q0, q1 = initial and final joint position.
%       qdot0, qdot1 = initial and final joint velocity.
%       tf = total movement time.
%       q  = vector of joint positions, equally spaced in time.

function q = cubic(q0,q1,qdot0,qdot1,tf)

a = zeros(4,1);

% Generate coefficients of cubic polynomial.
a(1) = q0;
a(2) = qdot0;
a(3) =  (3 * (q1-q0) - tf * (2 * qdot0 + qdot1)) / tf^2;
a(4) = (-2 * (q1-q0) + tf * (qdot0 + qdot1)) / tf^3;

% Divide time into 100 intervals.
dt = tf/100;
% This Matlab trick creates an array from 0 to tf with increment dt.
t  = 0:dt:tf;

% Generate joint trajectory.
q = a(1) + t .* (a(2) + t .* (a(3) + t * a(4)));

% Plot the results.  Control axis spacing.
plot(q);
axis([0 100 -20 10]);

% Save the plot for printing on a postscript printer.
print -deps2 cubic.eps
```

Bibliography

[1] Paul, R.P., "Manipulator Cartesian path control," *IEEE Trans. Systems, Man, Cybernetics*, vol. SMC-9, pp. 702-711, 1979.

[2] Taylor, R.H., "Planning and execution of straight-line manipulator trajectories," *IBM Journal of Research and Development*, vol. 23, pp. 424-436, 1979.