CHAPTER 6

Manipulator dynamics

6.1 INTRODUCTION

Our study of manipulators so far has focused on kinematic considerations only. We have studied static positions, static forces, and velocities; but we have never considered *the forces required to cause motion.* In this chapter, we consider the equations of motion for a manipulator—the way in which motion of the manipulator arises from torques applied by the actuators or from external forces applied to the manipulator.

Dynamics of mechanisms is a field in which many books have been written. Indeed, one can spend years studying the field. Obviously, we cannot cover the material in the completeness it deserves. However, certain formulations of the dynamics problem seem particularly well suited to application to manipulators. In particular, methods which make use of the serial-chain nature of manipulators are natural candidates for our study.

There are two problems related to the dynamics of a manipulator that we wish to solve. In the first problem, we are given a trajectory point, Θ, $\dot{\Theta}$, and $\ddot{\Theta}$, and we wish to find the required vector of joint torques, τ. This formulation of dynamics is useful for the problem of controlling the manipulator (Chapter 10). The second problem is to calculate how the mechanism will move under application of a set of joint torques. That is, given a torque vector, τ, calculate the resulting motion of the manipulator, Θ, $\dot{\Theta}$, and $\ddot{\Theta}$. This is useful for simulating the manipulator.

6.2 ACCELERATION OF A RIGID BODY

We now extend our analysis of rigid-body motion to the case of accelerations. At any instant, the linear and angular velocity vectors have derivatives that are called the linear and angular accelerations, respectively. That is,

$$ {}^{B}\dot{V}_{Q} = \frac{d}{dt}\,{}^{B}V_{Q} = \lim_{\Delta t \to o} \frac{{}^{B}V_{Q}(t+\Delta t) - {}^{B}V_{Q}(t)}{\Delta t}, \tag{6.1} $$

and

$$ {}^{A}\dot{\Omega}_{B} = \frac{d}{dt}\,{}^{A}\Omega_{B} = \lim_{\Delta t \to o} \frac{{}^{A}\Omega_{B}(t+\Delta t) - {}^{A}\Omega_{B}(t)}{\Delta t}. \tag{6.2} $$

As with velocities, when the reference frame of the differentiation is understood to be some universal reference frame, $\{U\}$, we will use the notation

$$ \dot{v}_{A} = {}^{U}\dot{V}_{AORG} \tag{6.3} $$

and

$$ \dot{\omega}_{A} = {}^{U}\dot{\Omega}_{A}. \tag{6.4} $$

Linear acceleration

We start by restating (5.12), an important result from Chapter 5, which describes the velocity of a vector ${}^{B}Q$ as seen from frame $\{A\}$ when the origins are coincident:

$$ {}^{A}V_{Q} = {}^{A}_{B}R\,{}^{B}V_{Q} + {}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}Q. \tag{6.5} $$

The left-hand side of this equation describes how ${}^{A}Q$ is changing in time. So, because origins are coincident, we could rewrite (6.5) as

$$ \frac{d}{dt}({}^{A}_{B}R\,{}^{B}Q) = {}^{A}_{B}R\,{}^{B}V_{Q} + {}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}Q. \tag{6.6} $$

This form of the equation will be useful when deriving the corresponding acceleration equation.

By differentiating (6.5), we can derive expressions for the acceleration of ${}^{B}Q$ as viewed from $\{A\}$ when the origins of $\{A\}$ and $\{B\}$ coincide:

$$ {}^{A}\dot{V}_{Q} = \frac{d}{dt}({}^{A}_{B}R\,{}^{B}V_{Q}) + {}^{A}\dot{\Omega}_{B} \times {}^{A}_{B}R\,{}^{B}Q + {}^{A}\Omega_{B} \times \frac{d}{dt}({}^{A}_{B}R\,{}^{B}Q). \tag{6.7} $$

Now we apply (6.6) twice—once to the first term, and once to the last term. The right-hand side of equation (6.7) becomes

$$ \begin{aligned} {}^{A}_{B}R\,{}^{B}\dot{V}_{Q} \;&+\; {}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}V_{Q} + {}^{A}\dot{\Omega}_{B} \times {}^{A}_{B}R\,{}^{B}Q \\ &+\; {}^{A}\Omega_{B} \times ({}^{A}_{B}R\,{}^{B}V_{Q} + {}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}Q). \end{aligned} \tag{6.8} $$

Combining two terms, we get

$$ {}^{A}_{B}R\,{}^{B}\dot{V}_{Q} + 2{}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}V_{Q} + {}^{A}\dot{\Omega}_{B} \times {}^{A}_{B}R\,{}^{B}Q + {}^{A}\Omega_{B} \times ({}^{A}\Omega_{B} \times {}^{A}_{B}R\,{}^{B}Q). \tag{6.9} $$

Finally, to generalize to the case in which the origins are not coincident, we add one term which gives the linear acceleration of the origin of $\{B\}$, resulting in the final general formula:

$$ {}^A\dot{V}_{BORG} + {}^A_B R\, {}^B\dot{V}_Q + 2{}^A\Omega_B \times {}^A_B R\, {}^B V_Q + {}^A\dot{\Omega}_B \times {}^A_B R\, {}^B Q + {}^A\Omega_B \times ({}^A\Omega_B \times {}^A_B R\, {}^B Q). \tag{6.10} $$

A particular case that is worth pointing out is when ${}^B Q$ is constant, or

$$ {}^B V_Q = {}^B\dot{V}_Q = 0. \tag{6.11} $$

In this case, (6.10) simplifies to

$$ {}^A\dot{V}_Q = {}^A\dot{V}_{BORG} + {}^A\Omega_B \times ({}^A\Omega_B \times {}^A_B R\, {}^B Q) + {}^A\dot{\Omega}_B \times {}^A_B R\, {}^B Q. \tag{6.12} $$

We will use this result in calculating the linear acceleration of the links of a manipulator with rotational joints. When a prismatic joint is present, the more general form of (6.10) will be used.

Angular acceleration

Consider the case in which $\{B\}$ is rotating relative to $\{A\}$ with ${}^A\Omega_B$ and $\{C\}$ is rotating relative to $\{B\}$ with ${}^B\Omega_C$. To calculate ${}^A\Omega_C$, we sum the vectors in frame $\{A\}$:

$$ {}^A\Omega_C = {}^A\Omega_B + {}^A_B R\, {}^B\Omega_C. \tag{6.13} $$

By differentiating, we obtain

$$ {}^A\dot{\Omega}_C = {}^A\dot{\Omega}_B + \frac{d}{dt}({}^A_B R\, {}^B\Omega_C). \tag{6.14} $$

Now, applying (6.6) to the last term of (6.14), we get

$$ {}^A\dot{\Omega}_C = {}^A\dot{\Omega}_B + {}^A_B R\, {}^B\dot{\Omega}_C + {}^A\Omega_B \times {}^A_B R\, {}^B\Omega_C. \tag{6.15} $$

We will use this result to calculate the angular acceleration of the links of a manipulator.

6.3 MASS DISTRIBUTION

In systems with a single degree of freedom, we often talk about the mass of a rigid body. In the case of rotational motion about a single axis, the notion of the *moment of inertia* is a familiar one. For a rigid body that is free to move in three dimensions, there are infinitely many possible rotation axes. In the case of rotation about an arbitrary axis, we need a complete way of characterizing the mass distribution of a rigid body. Here, we introduce the **inertia tensor**, which, for our purposes, can be thought of as a generalization of the scalar moment of inertia of an object.

We shall now define a set of quantities that give information about the distribution of mass of a rigid body relative to a reference frame. Figure 6.1 shows a rigid body with an attached frame. Inertia tensors can be defined relative to any frame, but we will always consider the case of an inertia tensor defined for a frame

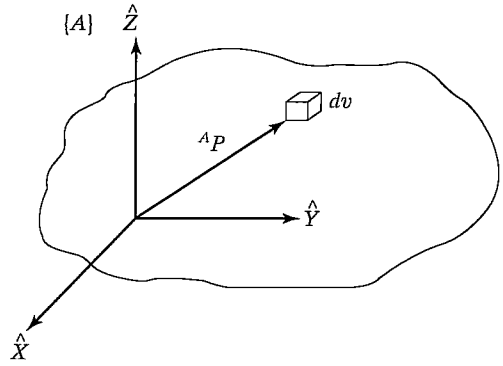

FIGURE 6.1: The inertia tensor of an object describes the object's mass distribution. Here, the vector ${}^{A}P$ locates the differential volume element, dv.

attached to the rigid body. Where it is important, we will indicate, with a leading superscript, the frame of reference of a given inertia tensor. The inertia tensor relative to frame $\{A\}$ is expressed in the matrix form as the 3×3 matrix

$$ {}^{A}I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}, \tag{6.16} $$

where the scalar elements are given by

$$ \begin{aligned} I_{xx} &= \iiint_V (y^2 + z^2)\rho dv, \\ I_{yy} &= \iiint_V (x^2 + z^2)\rho dv, \\ I_{zz} &= \iiint_V (x^2 + y^2)\rho dv, \\ I_{xy} &= \iiint_V xy\rho dv, \\ I_{xz} &= \iiint_V xz\rho dv, \\ I_{yz} &= \iiint_V yz\rho dv, \end{aligned} \tag{6.17} $$

in which the rigid body is composed of differential volume elements, dv, containing material of density ρ. Each volume element is located with a vector, ${}^{A}P = [xyz]^T$, as shown in Fig. 6.1.

The elements I_{xx}, I_{yy}, and I_{zz} are called the **mass moments of inertia**. Note that, in each case, we are integrating the mass elements, ρdv, times the squares of the perpendicular distances from the corresponding axis. The elements with mixed indices are called the **mass products of inertia**. This set of six independent quantities

will, for a given body, depend on the position and orientation of the frame in which they are defined. If we are free to choose the orientation of the reference frame, it is possible to cause the products of inertia to be zero. The axes of the reference frame when so aligned are called the **principal axes** and the corresponding mass moments are the **principal moments** of inertia.

EXAMPLE 6.1

Find the inertia tensor for the rectangular body of uniform density ρ with respect to the coordinate system shown in Fig. 6.2.

First, we compute I_{xx}. Using volume element $dv = dx\,dy\,dz$, we get

$$\begin{aligned} I_{xx} &= \int_0^h \int_0^l \int_0^\omega (y^2 + z^2)\rho\, dx\, dy\, dz \\ &= \int_0^h \int_0^l (y^2 + z^2)\omega\rho dy\, dz \\ &= \int_0^h \left(\frac{l^3}{3} + z^2 l\right)\omega\rho dz \\ &= \left(\frac{hl^3\omega}{3} + \frac{h^3 l\omega}{3}\right)\rho \\ &= \frac{m}{3}(l^2 + h^2), \end{aligned} \tag{6.18}$$

where m is the total mass of the body. Permuting the terms, we can get I_{yy} and I_{zz} by inspection:

$$I_{yy} = \frac{m}{3}(\omega^2 + h^2) \tag{6.19}$$

and

$$I_{zz} = \frac{m}{3}(l^2 + \omega^2). \tag{6.20}$$

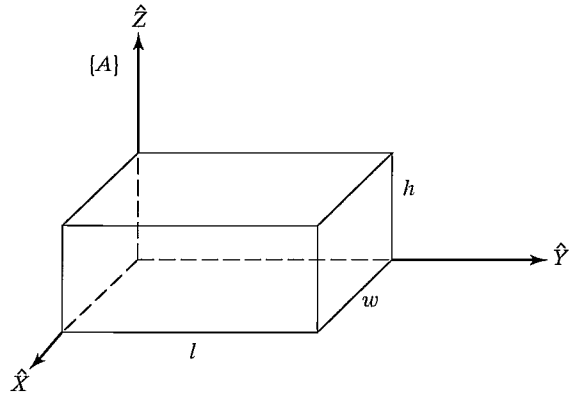

FIGURE 6.2: A body of uniform density.

We next compute I_{xy}:

$$\begin{aligned} I_{xy} &= \int_0^h \int_0^l \int_0^\omega xy\rho \, dx \, dy \, dz \\ &= \int_0^h \int_0^l \frac{\omega^2}{2} y\rho \, dy \, dz \\ &= \int_0^h \frac{\omega^2 l^2}{4} \rho dz \\ &= \frac{m}{4}\omega l. \end{aligned} \tag{6.21}$$

Permuting the terms, we get

$$I_{xz} = \frac{m}{4}h\omega \tag{6.22}$$

and

$$I_{yz} = \frac{m}{4}hl. \tag{6.23}$$

Hence, the inertia tensor for this object is

$${}^A I = \begin{bmatrix} \frac{m}{3}(l^2+h^2) & -\frac{m}{4}\omega l & -\frac{m}{4}h\omega \\ -\frac{m}{4}\omega l & \frac{m}{3}(\omega^2+h^2) & -\frac{m}{4}hl \\ -\frac{m}{4}h\omega & -\frac{m}{4}hl & \frac{m}{3}(l^2+\omega^2) \end{bmatrix}. \tag{6.24}$$

As noted, the inertia tensor is a function of the location and orientation of the reference frame. A well-known result, the **parallel-axis theorem**, is one way of computing how the inertia tensor changes under *translations* of the reference coordinate system. The parallel-axis theorem relates the inertia tensor in a frame with origin at the center of mass to the inertia tensor with respect to another reference frame. Where $\{C\}$ is located at the center of mass of the body, and $\{A\}$ is an arbitrarily translated frame, the theorem can be stated [1] as

$$\begin{aligned} {}^A I_{zz} &= {}^C I_{zz} + m(x_c^2 + y_c^2), \\ {}^A I_{xy} &= {}^C I_{xy} - m x_c y_c, \end{aligned} \tag{6.25}$$

where $P_c = [x_c, y_c, z_c]^T$ locates the center of mass relative to $\{A\}$. The remaining moments and products of inertia are computed from permutations of x,y, and z in (6.25). The theorem may be stated in vector–matrix form as

$${}^A I = {}^C I + m[P_c^T P_c I_3 - P_c P_c^T], \tag{6.26}$$

where I_3 is the 3×3 identity matrix.

EXAMPLE 6.2

Find the inertia tensor for the same solid body described for Example 6.1 when it is described in a coordinate system with origin at the body's center of mass.

We can apply the parallel-axis theorem, (6.25), where

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \omega \\ l \\ h \end{bmatrix}.$$

Next, we find

$$\begin{aligned} {}^{C}I_{zz} &= \frac{m}{12}(\omega^2 + l^2), \\ {}^{C}I_{xy} &= 0. \end{aligned} \tag{6.27}$$

The other elements are found by symmetry. The resulting inertia tensor written in the frame at the center of mass is

$${}^{C}I = \begin{bmatrix} \frac{m}{12}(h^2 + l^2) & 0 & 0 \\ 0 & \frac{m}{12}(\omega^2 + h^2) & 0 \\ 0 & 0 & \frac{m}{12}(l^2 + \omega^2) \end{bmatrix}. \tag{6.28}$$

The result is diagonal, so frame $\{C\}$ must represent the principal axes of this body.

Some additional facts about inertia tensors are as follows:

1. If two axes of the reference frame form a plane of symmetry for the mass distribution of the body, the products of inertia having as an index the coordinate that is normal to the plane of symmetry will be zero.
2. Moments of inertia must always be positive. Products of inertia may have either sign.
3. The sum of the three moments of inertia is invariant under orientation changes in the reference frame.
4. The eigenvalues of an inertia tensor are the principal moments for the body. The associated eigenvectors are the principal axes.

Most manipulators have links whose geometry and composition are somewhat complex, so that the application of (6.17) is difficult in practice. A pragmatic option is actually to measure rather than to calculate the moment of inertia of each link by using a measuring device (e.g., an *inertia pendulum*).

6.4 NEWTON'S EQUATION, EULER'S EQUATION

We will consider each link of a manipulator as a rigid body. If we know the location of the center of mass and the inertia tensor of the link, then its mass distribution is completely characterized. In order to move the links, we must accelerate and decelerate them. The forces required for such motion are a function of the acceleration desired and of the mass distribution of the links. Newton's equation, along with its rotational analog, Euler's equation, describes how forces, inertias, and accelerations relate.

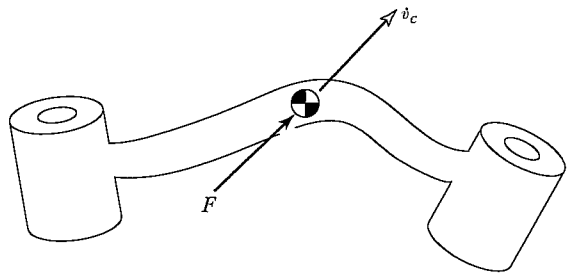

FIGURE 6.3: A force F acting at the center of mass of a body causes the body to accelerate at $\dot{v}_C$.

Newton's equation

Figure 6.3 shows a rigid body whose center of mass is accelerating with acceleration $\dot{v}_C$. In such a situation, the force, F, acting at the center of mass and causing this acceleration is given by Newton's equation

$$F = m\dot{v}_C, \tag{6.29}$$

where m is the total mass of the body.

Euler's equation

Figure 6.4 shows a rigid body rotating with angular velocity ω and with angular acceleration $\dot{\omega}$. In such a situation, the moment N, which must be acting on the body to cause this motion, is given by Euler's equation

$$N = {}^C I\dot{\omega} + \omega \times {}^C I\omega, \tag{6.30}$$

where ${}^C I$ is the inertia tensor of the body written in a frame, $\{C\}$, whose origin is located at the center of mass.

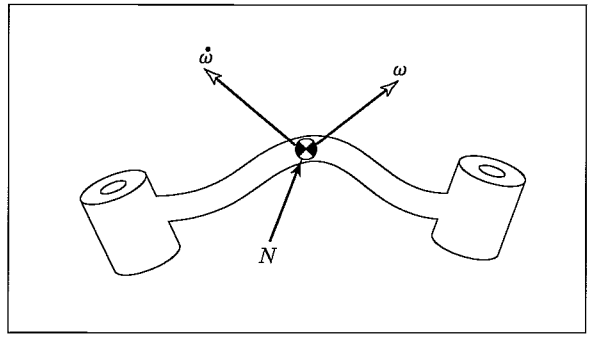

FIGURE 6.4: A moment N is acting on a body, and the body is rotating with velocity ω and accelerating at $\dot{\omega}$.

6.5 ITERATIVE NEWTON–EULER DYNAMIC FORMULATION

We now consider the problem of computing the torques that correspond to a given trajectory of a manipulator. We assume we know the position, velocity, and acceleration of the joints, $(\Theta, \dot{\Theta}, \ddot{\Theta})$. With this knowledge, and with knowledge of the kinematics and the mass-distribution information of the robot, we can calculate the joint torques required to cause this motion. The algorithm presented is based upon the method published by Luh, Walker, and Paul in [2].

Outward iterations to compute velocities and accelerations

In order to compute inertial forces acting on the links, it is necessary to compute the rotational velocity and linear and rotational acceleration of the center of mass of each link of the manipulator at any given instant. These computations will be done in an iterative way, starting with link 1 and moving successively, link by link, *outward* to link n.

The "propagation" of rotational velocity from link to link was discussed in Chapter 5 and is given (for joint $i+1$ rotational) by

$$ {}^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\,{}^{i}\omega_i + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}. \tag{6.31} $$

From (6.15), we obtain the equation for transforming angular acceleration from one link to the next:

$$ {}^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\,{}^{i}\dot{\omega}_i + {}^{i+1}_{i}R\,{}^{i}\omega_i \times \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}. \tag{6.32} $$

When joint $i+1$ is prismatic, this simplifies to

$$ {}^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\,{}^{i}\omega_i. \tag{6.33} $$

The linear acceleration of each link-frame origin is obtained by the application of (6.12):

$$ {}^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R[{}^{i}\omega_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i], \tag{6.34} $$

For prismatic joint $i+1$, (6.34) becomes (from (6.10))

$$ \begin{aligned} {}^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R({}^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i) \\ + 2{}^{i+1}\omega_{i+1} \times \dot{d}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} + \ddot{d}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}. \end{aligned} \tag{6.35} $$

We also will need the linear acceleration of the center of mass of each link, which also can be found by applying (6.12):

$$ {}^{i}\dot{v}_{C_i} = {}^{i}\dot{\omega}_i \times {}^{i}P_{C_i} + {}^{i}\omega_i \times ({}^{i}\omega_i + {}^{i}P_{C_i}) + {}^{i}\dot{v}_i, \tag{6.36} $$

Here, we imagine a frame, $\{C_i\}$, attached to each link, having its origin located at the center of mass of the link and having the same orientation as the link frame, $\{i\}$. Equation (6.36) doesn't involve joint motion at all and so is valid for joint $i+1$, regardless of whether it is revolute or prismatic.

Note that the application of the equations to link 1 is especially simple, because ${}^{0}\omega_0 = {}^{0}\dot{\omega}_0 = 0$.

The force and torque acting on a link

Having computed the linear and angular accelerations of the mass center of each link, we can apply the Newton–Euler equations (Section 6.4) to compute the inertial force and torque acting at the center of mass of each link. Thus we have

$$F_i = m\dot{v}_{C_i},$$
$$N_i = {}^{C_i}I\dot{\omega}_i + \omega_i \times {}^{C_i}I\omega_i, \tag{6.37}$$

where $\{C_i\}$ has its origin at the center of mass of the link and has the same orientation as the link frame, $\{i\}$.

Inward iterations to compute forces and torques

Having computed the forces and torques acting on each link, we now need to calculate the joint torques that will result in these net forces and torques being applied to each link.

We can do this by writing a force-balance and moment-balance equation based on a free-body diagram of a typical link. (See Fig. 6.5.) Each link has forces and torques exerted on it by its neighbors and in addition experiences an inertial force and torque. In Chapter 5, we defined special symbols for the force and torque exerted by a neighbor link, which we repeat here:

f_i = force exerted on link i by link $i-1$,
n_i = torque exerted on link i by link $i-1$.

By summing the forces acting on link i, we arrive at the force-balance relationship:

$${}^iF_i = {}^if_i - {}^i_{i+1}R\,{}^{i+1}f_{i+1}. \tag{6.38}$$

By summing torques about the center of mass and setting them equal to zero, we arrive at the torque-balance equation:

$${}^iN_i = {}^in_i - {}^in_{i+1} + (-{}^iP_{C_i}) \times {}^if_i - ({}^iP_{i+1} - {}^iP_{C_i}) \times {}^if_{i+1}. \tag{6.39}$$

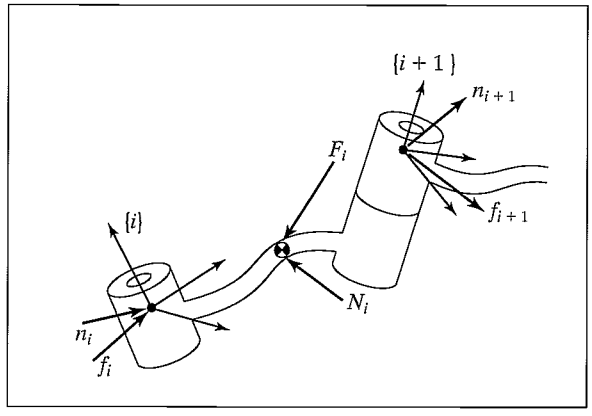

FIGURE 6.5: The force balance, including inertial forces, for a single manipulator link.

Using the result from the force-balance relation (6.38) and adding a few rotation matrices, we can write (6.39) as

$$ {}^{i}N_i = {}^{i}n_i - {}^{i}_{i+1}R\,{}^{i+1}n_{i+1} - {}^{i}P_{C_i} \times {}^{i}F_i - {}^{i}P_{i+1} \times {}^{i}_{i+1}R\,{}^{i+1}f_{i+1}. \tag{6.40} $$

Finally, we can rearrange the force and torque equations so that they appear as iterative relationships from higher numbered neighbor to lower numbered neighbor:

$$ {}^{i}f_i = {}^{i}_{i+1}R\,{}^{i+1}f_{i+1} + {}^{i}F_i, \tag{6.41} $$

$$ {}^{i}n_i = {}^{i}N_i + {}^{i}_{i+1}R\,{}^{i+1}n_{i+1} + {}^{i}P_{C_i} \times {}^{i}F_i + {}^{i}P_{i+1} \times {}^{i}_{i+1}R\,{}^{i+1}f_{i+1}. \tag{6.42} $$

These equations are evaluated link by link, starting from link n and working inward toward the base of the robot. These *inward force iterations* are analogous to the static force iterations introduced in Chapter 5, except that inertial forces and torques are now considered at each link.

As in the static case, the required joint torques are found by taking the $\hat{Z}$ component of the torque applied by one link on its neighbor:

$$ \tau_i = {}^{i}n_i^T\,{}^{i}\hat{Z}_i. \tag{6.43} $$

For joint i prismatic, we use

$$ \tau_i = {}^{i}f_i^T\,{}^{i}\hat{Z}_i, \tag{6.44} $$

where we have used the symbol τ for a linear actuator force.

Note that, for a robot moving in free space, ${}^{N+1}f_{N+1}$ and ${}^{N+1}n_{N+1}$ are set equal to zero, and so the first application of the equations for link n is very simple. If the robot is in contact with the environment, the forces and torques due to this contact can be included in the force balance by having nonzero ${}^{N+1}f_{N+1}$ and ${}^{N+1}n_{N+1}$.

The iterative Newton–Euler dynamics algorithm

The complete algorithm for computing joint torques from the motion of the joints is composed of two parts. First, link velocities and accelerations are iteratively computed from link 1 out to link n and the Newton–Euler equations are applied to each link. Second, forces and torques of interaction and joint actuator torques are computed recursively from link n back to link 1. The equations are summarized next for the case of all joints rotational:

Outward iterations: $i : 0 \rightarrow 5$

$$^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\,{}^{i}\omega_i + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}, \tag{6.45}$$

$$^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\,{}^{i}\dot{\omega}_i + {}^{i+1}_{i}R\,{}^{i}\omega_i \times \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1}, \tag{6.46}$$

$$^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R(^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i), \tag{6.47}$$

$$\begin{aligned} ^{i+1}\dot{v}_{C_{i+1}} &= {}^{i+1}\dot{\omega}_{i+1} \times {}^{i+1}P_{C_{i+1}} \\ &\quad + {}^{i+1}\omega_{i+1} \times ({}^{i+1}\omega_{i+1} \times {}^{i+1}P_{C_{i+1}}) + {}^{i+1}\dot{v}_{i+1}, \end{aligned} \tag{6.48}$$

$$^{i+1}F_{i+1} = m_{i+1}\,{}^{i+1}\dot{v}_{C_{i+1}}, \tag{6.49}$$

$$^{i+1}N_{i+1} = {}^{C_{i+1}}I_{i+1}\,{}^{i+1}\dot{\omega}_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{C_{i+1}}I_{i+1}\,{}^{i+1}\omega_{i+1}. \tag{6.50}$$

Inward iterations: $i : 6 \rightarrow 1$

$$^{i}f_i = {}^{i}_{i+1}R\,{}^{i+1}f_{i+1} + {}^{i}F_i, \tag{6.51}$$

$$\begin{aligned} ^{i}n_i &= {}^{i}N_i + {}^{i}_{i+1}R\,{}^{i+1}n_{i+1} + {}^{i}P_{C_i} \times {}^{i}F_i \\ &\quad + {}^{i}P_{i+1} \times {}^{i}_{i+1}R\,{}^{i+1}f_{i+1}, \end{aligned} \tag{6.52}$$

$$\tau_i = {}^{i}n_i^T\,{}^{i}\hat{Z}_i. \tag{6.53}$$

Inclusion of gravity forces in the dynamics algorithm

The effect of gravity loading on the links can be included quite simply by setting ${}^{0}\dot{v}_0 = G$, where G has the magnitude of the gravity vector but points in the opposite direction. This is equivalent to saying that the base of the robot is accelerating upward with 1 g acceleration. This fictitious upward acceleration causes exactly the same effect on the links as gravity would. So, with no extra computational expense, the gravity effect is calculated.

6.6 ITERATIVE VS. CLOSED FORM

Equations (6.46) through (6.53) give a computational scheme whereby, given the joint positions, velocities, and accelerations, we can compute the required joint torques. As with our development of equations to compute the Jacobian in Chapter 5, these relations can be used in two ways: as a numerical computational algorithm, or as an algorithm used analytically to develop symbolic equations.

Use of the equations as a numerical computational algorithm is attractive because the equations apply to any robot. Once the inertia tensors, link masses, P_{C_i} vectors, and ${}^{i+1}_{i}R$ matrices are specified for a particular manipulator, the equations can be applied directly to compute the joint torques corresponding to any motion.

However, we often are interested in obtaining better insight into the structure of the equations. For example, what is the form of the gravity terms? How does the magnitude of the gravity effects compare with the magnitude of the inertial effects? To investigate these and other questions, it is often useful to write closed-form dynamic equations. These equations can be derived by applying the recursive

Newton–Euler equations symbolically to Θ, $\dot{\Theta}$, and $\ddot{\Theta}$. This is analogous to what we did in Chapter 5 to derive the symbolic form of the Jacobian.

6.7 AN EXAMPLE OF CLOSED-FORM DYNAMIC EQUATIONS

Here we compute the closed-form dynamic equations for the two-link planar manipulator shown in Fig. 6.6. For simplicity, we assume that the mass distribution is extremely simple: All mass exists as a point mass at the distal end of each link. These masses are m_1 and m_2.

First, we determine the values of the various quantities that will appear in the recursive Newton–Euler equations. The vectors that locate the center of mass for each link are

$$^1P_{C_1} = l_1\hat{X}_1,$$
$$^2P_{C_2} = l_2\hat{X}_2.$$

Because of the point-mass assumption, the inertia tensor written at the center of mass for each link is the zero matrix:

$$^{C_1}I_1 = 0,$$
$$^{C_2}I_2 = 0.$$

There are no forces acting on the end-effector, so we have

$$f_3 = 0,$$
$$n_3 = 0.$$

The base of the robot is not rotating; hence, we have

$$\omega_0 = 0,$$
$$\dot{\omega}_0 = 0.$$

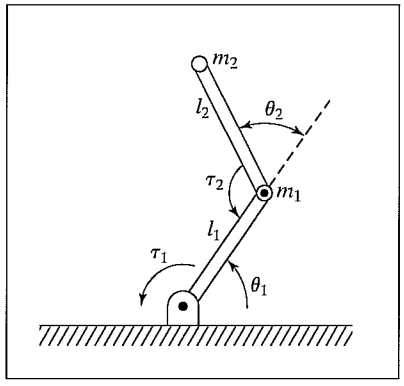

FIGURE 6.6: Two-link planar manipulator with point masses at distal ends of links.

To include gravity forces, we will use

$$^0\dot{v}_0 = g\hat{Y}_0.$$

The rotation between successive link frames is given by

$$^{i}_{i+1}R = \begin{bmatrix} c_{i+1} & -s_{i+1} & 0.0 \\ s_{i+1} & c_{i+1} & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix},$$

$$^{i+1}_{i}R = \begin{bmatrix} c_{i+1} & s_{i+1} & 0.0 \\ -s_{i+1} & c_{i+1} & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}.$$

We now apply equations (6.46) through (6.53).

The outward iterations for link 1 are as follows:

$$^1\omega_1 = \dot{\theta}_1\, ^1\hat{Z}_1 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix},$$

$$^1\dot{\omega}_1 = \ddot{\theta}_1\, ^1\hat{Z}_1 = \begin{bmatrix} 0 \\ 0 \\ \ddot{\theta}_1 \end{bmatrix},$$

$$^1\dot{v}_1 = \begin{bmatrix} c_1 & s_1 & 0 \\ -s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix} = \begin{bmatrix} gs_1 \\ gc_1 \\ 0 \end{bmatrix},$$

$$^1\dot{v}_{C_1} = \begin{bmatrix} 0 \\ l_1\ddot{\theta}_1 \\ 0 \end{bmatrix} + \begin{bmatrix} -l_1\dot{\theta}_1^2 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} gs_1 \\ gc_1 \\ 0 \end{bmatrix} = \begin{bmatrix} -l_1\dot{\theta}_1^2 + gs_1 \\ l_1\ddot{\theta}_1 + gc_1 \\ 0 \end{bmatrix},$$

$$^1F_1 = \begin{bmatrix} -m_1l_1\dot{\theta}_1^2 + m_1gs_1 \\ m_1l_1\ddot{\theta}_1 + m_1gc_1 \\ 0 \end{bmatrix},$$

$$^1N_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{6.54}$$

The outward iterations for link 2 are as follows:

$$^2\omega_2 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 + \dot{\theta}_2 \end{bmatrix},$$

$$^2\dot{\omega}_2 = \begin{bmatrix} 0 \\ 0 \\ \ddot{\theta}_1 + \ddot{\theta}_2 \end{bmatrix},$$

$$
{}^2\dot{v}_2 = \begin{bmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -l_1\dot{\theta}_1^2 + gs_1 \\ l_1\ddot{\theta}_1 + gc_1 \\ 0 \end{bmatrix} = \begin{bmatrix} l_1\ddot{\theta}_1 s_2 - l_1\dot{\theta}_1^2 c_2 + gs_{12} \\ l_1\ddot{\theta}_1 c_2 + l_1\dot{\theta}_1^2 s_2 + gc_{12} \\ 0 \end{bmatrix},
$$

$$
\begin{aligned}
{}^2\dot{v}_{C_2} = & \begin{bmatrix} 0 \\ l_2(\ddot{\theta}_1 + \ddot{\theta}_2) \\ 0 \end{bmatrix} + \begin{bmatrix} -l_2(\dot{\theta}_1 + \dot{\theta}_2)^2 \\ 0 \\ 0 \end{bmatrix} \\
& + \begin{bmatrix} l_1\ddot{\theta}_1 s_2 - l_1\dot{\theta}_1^2 c_2 + gs_{12} \\ l_1\ddot{\theta}_1 c_2 + l_1\dot{\theta}_1^2 s_2 + gc_{12} \\ 0 \end{bmatrix},
\end{aligned} \tag{6.55}
$$

$$
{}^2F_2 = \begin{bmatrix} m_2 l_1\ddot{\theta}_1 s_2 - m_2 l_1\dot{\theta}_1^2 c_2 + m_2 gs_{12} - m_2 l_2(\dot{\theta}_1 + \dot{\theta}_2)^2 \\ m_2 l_1\ddot{\theta}_1 c_2 + m_2 l_1\dot{\theta}_1^2 s_2 + m_2 gc_{12} + m_2 l_2(\ddot{\theta}_1 + \ddot{\theta}_2) \\ 0 \end{bmatrix},
$$

$$
{}^2N_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.
$$

The inward iterations for link 2 are as follows:

$$
\begin{aligned}
{}^2f_2 &= {}^2F_2, \\
{}^2n_2 &= \begin{bmatrix} 0 \\ 0 \\ m_2 l_1 l_2 c_2\ddot{\theta}_1 + m_2 l_1 l_2 s_2\dot{\theta}_1^2 + m_2 l_2 gc_{12} + m_2 l_2^2(\ddot{\theta}_1 + \ddot{\theta}_2) \end{bmatrix}.
\end{aligned} \tag{6.56}
$$

The inward iterations for link 1 are as follows:

$$
\begin{aligned}
{}^1f_1 = & \begin{bmatrix} c_2 & -s_2 & 0 \\ s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_2 l_1 s_2\ddot{\theta}_1 - m_2 l_1 c_2\dot{\theta}_1^2 + m_2 gs_{12} - m_2 l_2(\dot{\theta}_1 + \dot{\theta}_2)^2 \\ m_2 l_1 c_2\ddot{\theta}_1 + m_2 l_1 s_2\dot{\theta}_1^2 + m_2 gc_{12} + m_2 l_2(\ddot{\theta}_1 + \ddot{\theta}_2) \\ 0 \end{bmatrix} \\
& + \begin{bmatrix} -m_1 l_1\dot{\theta}_1^2 + m_1 gs_1 \\ m_1 l_1\ddot{\theta}_1 + m_1 gc_1 \\ 0 \end{bmatrix},
\end{aligned}
$$

$$
\begin{aligned}
{}^1n_1 = & \begin{bmatrix} 0 \\ 0 \\ m_2 l_1 l_2 c_2\ddot{\theta}_1 + m_2 l_1 l_2 s_2\dot{\theta}_1^2 + m_2 l_2 gc_{12} + m_2 l_2^2(\ddot{\theta}_1 + \ddot{\theta}_2) \end{bmatrix} \\
& + \begin{bmatrix} 0 \\ 0 \\ m_1 l_1^2\ddot{\theta}_1 + m_1 l_1 gc_1 \end{bmatrix} \\
& + \begin{bmatrix} 0 \\ 0 \\ m_2 l_1^2\ddot{\theta}_1 - m_2 l_1 l_2 s_2(\dot{\theta}_1 + \dot{\theta}_2)^2 + m_2 l_1 gs_2 s_{12} \\ +m_2 l_1 l_2 c_2(\ddot{\theta}_1 + \ddot{\theta}_2) + m_2 l_1 gc_2 c_{12} \end{bmatrix}.
\end{aligned} \tag{6.57}
$$

Extracting the $\hat{Z}$ components of the ${}^{i}n_i$, we find the joint torques:

$$\begin{aligned}\tau_1 &= m_2 l_2^2(\ddot{\theta}_1 + \ddot{\theta}_2) + m_2 l_1 l_2 c_2(2\ddot{\theta}_1 + \ddot{\theta}_2) + (m_1 + m_2) l_1^2 \ddot{\theta}_1 - m_2 l_1 l_2 s_2 \dot{\theta}_2^2 \\ &\quad -2m_2 l_1 l_2 s_2 \dot{\theta}_1 \dot{\theta}_2 + m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1, \\ \tau_2 &= m_2 l_1 l_2 c_2 \ddot{\theta}_1 + m_2 l_1 l_2 s_2 \dot{\theta}_1^2 + m_2 l_2 g c_{12} + m_2 l_2^2(\ddot{\theta}_1 + \ddot{\theta}_2). \end{aligned} \tag{6.58}$$

Equations (6.58) give expressions for the torque at the actuators as a function of joint position, velocity, and acceleration. Note that these rather complex functions arose from one of the simplest manipulators imaginable. Obviously, the closed-form equations for a manipulator with six degrees of freedom will be quite complex.

6.8 THE STRUCTURE OF A MANIPULATOR'S DYNAMIC EQUATIONS

It is often convenient to express the dynamic equations of a manipulator in a single equation that hides some of the details, but shows some of the structure of the equations.

The state-space equation

When the Newton–Euler equations are evaluated symbolically for any manipulator, they yield a dynamic equation that can be written in the form

$$\tau = M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta), \tag{6.59}$$

where $M(\Theta)$ is the $n \times n$ **mass matrix** of the manipulator, $V(\Theta, \dot{\Theta})$ is an $n \times 1$ vector of centrifugal and Coriolis terms, and $G(\Theta)$ is an $n \times 1$ vector of gravity terms. We use the term **state-space equation** because the term $V(\Theta, \dot{\Theta})$, appearing in (6.59), has both position and velocity dependence [3].

Each element of $M(\Theta)$ and $G(\Theta)$ is a complex function that depends on Θ, the position of all the joints of the manipulator. Each element of $V(\Theta, \dot{\Theta})$ is a complex function of both Θ and $\dot{\Theta}$.

We may separate the various types of terms appearing in the dynamic equations and form the mass matrix of the manipulator, the centrifugal and Coriolis vector, and the gravity vector.

EXAMPLE 6.3

Give $M(\Theta)$, $V(\Theta, \dot{\Theta})$, and $G(\Theta)$ for the manipulator of Section 6.7.

Equation (6.59) defines the manipulator mass matrix, $M(\Theta)$; it is composed of all those terms which multiply $\ddot{\Theta}$ and is a function of Θ. Therefore, we have

$$M(\Theta) = \begin{bmatrix} l_2^2 m_2 + 2l_1 l_2 m_2 c_2 + l_1^2(m_1 + m_2) & l_2^2 m_2 + l_1 l_2 m_2 c_2 \\ l_2^2 m_2 + l_1 l_2 m_2 c_2 & l_2^2 m_2 \end{bmatrix}. \tag{6.60}$$

Any manipulator mass matrix is symmetric and positive definite, and is, therefore, always invertible.

The velocity term, $V(\Theta, \dot{\Theta})$, contains all those terms that have any dependence on joint velocity. Thus, we obtain

$$V(\Theta, \dot{\Theta}) = \begin{bmatrix} -m_2 l_1 l_2 s_2 \dot{\theta}_2^2 - 2m_2 l_1 l_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ m_2 l_1 l_2 s_2 \dot{\theta}_1^2 \end{bmatrix}. \tag{6.61}$$

A term like $-m_2 l_1 l_2 s_2 \dot{\theta}_2^2$ is caused by a **centrifugal force**, and is recognized as such because it depends on the square of a joint velocity. A term such as $-2m_2 l_1 l_2 s_2 \dot{\theta}_1 \dot{\theta}_2$ is caused by a **Coriolis force** and will always contain the product of two different joint velocities.

The gravity term, $G(\Theta)$, contains all those terms in which the gravitational constant, g, appears. Therefore, we have

$$G(\Theta) = \begin{bmatrix} m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1 \\ m_2 l_2 g c_{12} \end{bmatrix}. \tag{6.62}$$

Note that the gravity term depends only on Θ and not on its derivatives.

The configuration-space equation

By writing the velocity-dependent term, $V(\Theta, \dot{\Theta})$, in a different form, we can write the dynamic equations as

$$\tau = M(\Theta)\ddot{\Theta} + B(\Theta)[\dot{\Theta}\dot{\Theta}] + C(\Theta)[\dot{\Theta}^2] + G(\Theta), \tag{6.63}$$

where $B(\Theta)$ is a matrix of dimensions $n \times n(n-1)/2$ of Coriolis coefficients, $[\dot{\Theta}\dot{\Theta}]$ is an $n(n-1)/2 \times 1$ vector of joint velocity products given by

$$[\dot{\Theta}\dot{\Theta}] = [\dot{\theta}_1\dot{\theta}_2 \ \dot{\theta}_1\dot{\theta}_3 \ \ldots \ \dot{\theta}_{n-1}\dot{\theta}_n]^T, \tag{6.64}$$

$C(\Theta)$ is an $n \times n$ matrix of centrifugal coefficients, and $[\dot{\Theta}^2]$ is an $n \times 1$ vector given by

$$[\dot{\theta}_1^2 \ \dot{\theta}_2^2 \ldots \dot{\theta}_n^2]^T. \tag{6.65}$$

We will call (6.63) the **configuration-space equation**, because the matrices are functions only of manipulator position [3].

In this form of the dynamic equations, the complexity of the computation is seen to be in the form of computing various parameters which are a function of only the manipulator position, Θ. This is important in applications (such as computer control of a manipulator) in which the dynamic equations must be updated as the manipulator moves. (Equation (6.63) gives a form in which parameters are a function of joint position only and can be updated at a rate related to how fast the manipulator is changing configuration.) We will consider this form again with regard to the problem of manipulator control in Chapter 10.

EXAMPLE 6.4

Give $B(\Theta)$ and $C(\Theta)$ (from (6.63)) for the manipulator of Section 6.7.

For this simple two-link manipulator, we have

$$[\dot{\Theta}\Theta] = [\dot{\theta}_1\dot{\theta}_2],$$

$$[\dot{\Theta}^2] = \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{bmatrix}. \tag{6.66}$$

So we see that

$$B(\Theta) = \begin{bmatrix} -2m_2 l_1 l_2 s_2 \\ 0 \end{bmatrix} \tag{6.67}$$

and

$$C(\Theta) = \begin{bmatrix} 0 & -m_2 l_1 l_2 s_2 \\ m_2 l_1 l_2 s_2 & 0 \end{bmatrix}. \tag{6.68}$$

6.9 LAGRANGIAN FORMULATION OF MANIPULATOR DYNAMICS

The Newton–Euler approach is based on the elementary dynamic formulas (6.29) and (6.30) and on an analysis of forces and moments of constraint acting between the links. As an alternative to the Newton–Euler method, in this section we briefly introduce the **Lagrangian dynamic formulation**. Whereas the Newton–Euler formulation might be said to be a "force balance" approach to dynamics, the Lagrangian formulation is an "energy-based" approach to dynamics. Of course, for the same manipulator, both will give the same equations of motion. Our statement of Lagrangian dynamics will be brief and somewhat specialized to the case of a serial-chain mechanical manipulator with rigid links. For a more complete and general reference, see [4].

We start by developing an expression for the kinetic energy of a manipulator. The kinetic energy of the ith link, k_i, can be expressed as

$$k_i = \tfrac{1}{2} m_i v_{C_i}^T v_{C_i} + \tfrac{1}{2} {}^i\omega_i^T {}^{C_i}I_i {}^i\omega_i, \tag{6.69}$$

where the first term is kinetic energy due to linear velocity of the link's center of mass and the second term is kinetic energy due to angular velocity of the link. The total kinetic energy of the manipulator is the sum of the kinetic energy in the individual links—that is,

$$k = \sum_{i=1}^{n} k_i. \tag{6.70}$$

The v_{C_i} and ${}^i\omega_i$ in (6.69) are functions of Θ and $\dot{\Theta}$, so we see that the kinetic energy of a manipulator can be described by a scalar formula as a function of joint position and velocity, $k(\Theta, \dot{\Theta})$. In fact, the kinetic energy of a manipulator is given by

$$k(\Theta, \dot{\Theta}) = \tfrac{1}{2} \dot{\Theta}^T M(\Theta) \dot{\Theta}, \tag{6.71}$$

where $M(\Theta)$ is the $n \times n$ manipulator mass matrix already introduced in Section 6.8. An expression of the form of (6.71) is known as a **quadratic form** [5], since when expanded out, the resulting scalar equation is composed solely of terms whose dependence on the $\dot{\theta}_i$ is quadratic. Further, because the total kinetic energy must always be positive, the manipulator mass matrix must be a so-called **positive definite** matrix. Positive definite matrices are those having the property that their quadratic form is always a positive scalar. Equation (6.71) can be seen to be analogous to the familiar expression for the kinetic energy of a point mass:

$$k = \tfrac{1}{2} m v^2. \tag{6.72}$$

The fact that a manipulator mass matrix must be positive definite is analogous to the fact that a scalar mass is always a positive number.

The potential energy of the ith link, u_i, can be expressed as

$$u_i = -m_i\, {}^0g^T\, {}^0P_{C_i} + u_{ref_i}, \tag{6.73}$$

where ${}^0g$ is the 3×1 gravity vector, ${}^0P_{C_i}$ is the vector locating the center of mass of the ith link, and u_{ref_i} is a constant chosen so that the minimum value of u_i is zero.[1] The total potential energy stored in the manipulator is the sum of the potential energy in the individual links—that is,

$$u = \sum_{i=1}^{n} u_i. \tag{6.74}$$

Because the ${}^0P_{C_i}$ in (6.73) are functions of Θ, we see that the potential energy of a manipulator can be described by a scalar formula as a function of joint position, $u(\Theta)$.

The Lagrangian dynamic formulation provides a means of deriving the equations of motion from a scalar function called the **Lagrangian**, which is defined as the difference between the kinetic and potential energy of a mechanical system. In our notation, the Lagrangian of a manipulator is

$$\mathcal{L}(\Theta, \dot{\Theta}) = k(\Theta, \dot{\Theta}) - u(\Theta). \tag{6.75}$$

The equations of motion for the manipulator are then given by

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\Theta}} - \frac{\partial \mathcal{L}}{\partial \Theta} = \tau, \tag{6.76}$$

where τ is the $n \times 1$ vector of actuator torques. In the case of a manipulator, this equation becomes

$$\frac{d}{dt}\frac{\partial k}{\partial \dot{\Theta}} - \frac{\partial k}{\partial \Theta} + \frac{\partial u}{\partial \Theta} = \tau, \tag{6.77}$$

where the arguments of $k(\cdot)$ and $u(\cdot)$ have been dropped for brevity.

EXAMPLE 6.5

The links of an RP manipulator, shown in Fig. 6.7, have inertia tensors

$$\begin{aligned} {}^{C_1}I_1 &= \begin{bmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{bmatrix}, \\ {}^{C_2}I_2 &= \begin{bmatrix} I_{xx2} & 0 & 0 \\ 0 & I_{yy2} & 0 \\ 0 & 0 & I_{zz2} \end{bmatrix}, \end{aligned} \tag{6.78}$$

[1] Actually, only the partial derivative of the potential energy with respect to Θ will appear in the dynamics, so this constant is arbitrary. This corresponds to defining the potential energy relative to an arbitrary zero reference height.

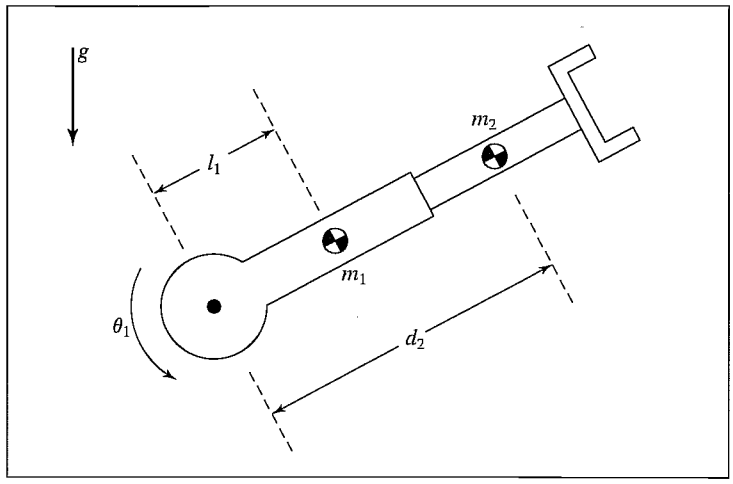

FIGURE 6.7: The RP manipulator of Example 6.5.

and total mass m_1 and m_2. As shown in Fig. 6.7, the center of mass of link 1 is located at a distance l_1 from the joint-1 axis, and the center of mass of link 2 is at the variable distance d_2 from the joint-1 axis. Use Lagrangian dynamics to determine the equation of motion for this manipulator.

Using (6.69), we write the kinetic energy of link 1 as

$$k_1 = \tfrac{1}{2} m_1 l_1^2 \dot{\theta}_1^2 + \tfrac{1}{2} I_{zz1} \dot{\theta}_1^2 \tag{6.79}$$

and the kinetic energy of link 2 as

$$k_2 = \tfrac{1}{2} m_2 (d_2^2 \dot{\theta}_1^2 + \dot{d}_2^2) + \tfrac{1}{2} I_{zz2} \dot{\theta}_1^2. \tag{6.80}$$

Hence, the total kinetic energy is given by

$$k(\Theta, \dot{\Theta}) = \tfrac{1}{2}(m_1 l_1^2 + I_{zz1} + I_{zz2} + m_2 d_2^2)\dot{\theta}_1^2 + \tfrac{1}{2} m_2 \dot{d}_2^2. \tag{6.81}$$

Using (6.73), we write the potential energy of link 1 as

$$u_1 = m_1 l_1 g \sin(\theta_1) + m_1 l_1 g \tag{6.82}$$

and the potential energy of link 2 as

$$u_2 = m_2 g d_2 \sin(\theta_1) + m_2 g d_{2max}, \tag{6.83}$$

where d_{2max} is the maximum extension of joint 2. Hence, the total potential energy is given by

$$u(\Theta) = g(m_1 l_1 + m_2 d_2) \sin(\theta_1) + m_1 l_1 g + m_2 g d_{2max}. \tag{6.84}$$

Next, we take partial derivatives as needed for (6.77):

$$\frac{\partial k}{\partial \dot{\Theta}} = \begin{bmatrix} (m_1 l_1^2 + I_{zz1} + I_{zz2} + m_2 d_2^2)\dot{\theta}_1 \\ m_2 d_2 \end{bmatrix}, \tag{6.85}$$

$$\frac{\partial k}{\partial \Theta} = \begin{bmatrix} 0 \\ m_2 d_2 \dot{\theta}_1^2 \end{bmatrix}, \tag{6.86}$$

$$\frac{\partial u}{\partial \Theta} = \begin{bmatrix} g(m_1 l_1 + m_2 d_2)\cos(\theta_1) \\ g m_2 \sin(\theta_1) \end{bmatrix}. \tag{6.87}$$

Finally, substituting into (6.77), we have

$$\begin{aligned} \tau_1 &= (m_1 l_1^2 + I_{zz1} + I_{zz2} + m_2 d_2^2)\ddot{\theta}_1 + 2m_2 d_2 \dot{\theta}_1 \dot{d}_2 \\ &\quad + (m_1 l_1 + m_2 d_2) g \cos(\theta_1), \\ \tau_2 &= m_2 \ddot{d}_2 - m_2 d_2 \dot{\theta}_1^2 + m_2 g \sin(\theta_1). \end{aligned} \tag{6.88}$$

From (6.89), we can see that

$$\begin{aligned} M(\Theta) &= \begin{bmatrix} (m_1 l_1^2 + I_{zz1} + I_{zz2} + m_2 d_2^2) & 0 \\ 0 & m_2 \end{bmatrix}, \\ V(\Theta, \dot{\Theta}) &= \begin{bmatrix} 2m_2 d_2 \dot{\theta}_1 \dot{d}_2 \\ -m_2 d_2 \dot{\theta}_1^2 \end{bmatrix}, \\ G(\Theta) &= \begin{bmatrix} (m_1 l_1 + m_2 d_2) g \cos(\theta_1) \\ m_2 g \sin(\theta_1) \end{bmatrix}. \end{aligned} \tag{6.89}$$

6.10 FORMULATING MANIPULATOR DYNAMICS IN CARTESIAN SPACE

Our dynamic equations have been developed in terms of the position and time derivatives of the manipulator joint angles, or in **joint space,** with the general form

$$\tau = M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta). \tag{6.90}$$

We developed this equation in joint space because we could use the serial-link nature of the mechanism to advantage in deriving the equations. In this section, we discuss the formulation of the dynamic equations that relate acceleration of the end-effector expressed in Cartesian space to Cartesian forces and moments acting at the end-effector.

The Cartesian state-space equation

As explained in Chapters 10 and 11, it might be desirable to express the dynamics of a manipulator with respect to Cartesian variables in the general form [6]

$$\mathcal{F} = M_x(\Theta)\ddot{\chi} + V_x(\Theta, \dot{\Theta}) + G_x(\Theta), \tag{6.91}$$

where $\mathcal{F}$ is a force–torque vector acting on the end-effector of the robot, and χ is an appropriate Cartesian vector representing position and orientation of the end-effector [7]. Analogous to the joint-space quantities, $M_x(\Theta)$ is the **Cartesian mass matrix**, $V_x(\Theta, \dot{\Theta})$ is a vector of velocity terms in Cartesian space, and $G_x(\Theta)$ is a vector of gravity terms in Cartesian space. Note that the fictitious forces acting on the end-effector, $\mathcal{F}$, could in fact be applied by the actuators at the joints by using the relationship

$$\tau = J^T(\Theta)\mathcal{F}, \tag{6.92}$$

where the Jacobian, $J(\Theta)$, is written in the same frame as $\mathcal{F}$ and $\ddot{\chi}$, usually the tool frame, $\{T\}$.

We can derive the relationship between the terms of (6.90) and those of (6.91) in the following way. First, we premultiply (6.90) by the inverse of the Jacobian transpose to obtain

$$J^{-T}\tau = J^{-T}M(\Theta)\ddot{\Theta} + J^{-T}V(\Theta, \dot{\Theta}) + J^{-T}G(\Theta), \tag{6.93}$$

or

$$\mathcal{F} = J^{-T}M(\Theta)\ddot{\Theta} + J^{-T}V(\Theta, \dot{\Theta}) + J^{-T}G(\Theta). \tag{6.94}$$

Next, we develop a relationship between joint space and Cartesian acceleration, starting with the definition of the Jacobian,

$$\dot{\chi} = J\dot{\Theta}, \tag{6.95}$$

and differentiating to obtain

$$\ddot{\chi} = \dot{J}\dot{\Theta} + J\ddot{\Theta}. \tag{6.96}$$

Solving (6.96) for joint-space acceleration leads to

$$\ddot{\Theta} = J^{-1}\ddot{\chi} - J^{-1}\dot{J}\dot{\Theta}. \tag{6.97}$$

Substituting (6.97) into (6.94), we have

$$\mathcal{F} = J^{-T}M(\Theta)J^{-1}\ddot{\chi} - J^{-T}M(\Theta)J^{-1}\dot{J}\dot{\Theta} + J^{-T}V(\Theta, \dot{\Theta}) + J^{-T}G(\Theta), \tag{6.98}$$

from which we derive the expressions for the terms in the Cartesian dynamics as

$$\begin{aligned} M_x(\Theta) &= J^{-T}(\Theta)M(\Theta)J^{-1}(\Theta), \\ V_x(\Theta, \dot{\Theta}) &= J^{-T}(\Theta)(V(\Theta, \dot{\Theta}) - M(\Theta)J^{-1}(\Theta)\dot{J}(\Theta)\dot{\Theta}), \\ G_x(\Theta) &= J^{-T}(\Theta)G(\Theta). \end{aligned} \tag{6.99}$$

Note that the Jacobian appearing in equations (6.100) is written in the same frames as $\mathcal{F}$ and χ in (6.91); the choice of this frame is arbitrary.[2] Note that, when the manipulator approaches a singularity, certain quantities in the Cartesian dynamics become infinite.

[2]Certain choices could facilitate computation.

EXAMPLE 6.6

Derive the Cartesian-space form of the dynamics for the two-link planar arm of Section 6.7. Write the dynamics in terms of a frame attached to the end of the second link.

For this manipulator, we have already obtained the dynamics (in Section 6.7) and the Jacobian (equation (5.66)), which we restate here:

$$J(\Theta) = \begin{bmatrix} l_1 s_2 & 0 \\ l_1 c_2 + l_2 & l_2 \end{bmatrix}. \tag{6.100}$$

First, compute the inverse Jacobian:

$$J^{-1}(\Theta) = \frac{1}{l_1 l_2 s_2} \begin{bmatrix} l_2 & 0 \\ -l_1 c_2 - l_2 & l_1 s_2 \end{bmatrix}. \tag{6.101}$$

Next, obtain the time derivative of the Jacobian:

$$\dot{J}(\Theta) = \begin{bmatrix} l_1 c_2 \dot{\theta}_2 & 0 \\ -l_1 s_2 \theta_2 & 0 \end{bmatrix}. \tag{6.102}$$

Using (6.100) and the results of Section 6.7, we get

$$M_x(\Theta) = \begin{bmatrix} m_2 + \frac{m_1}{s_2^2} & 0 \\ 0 & m_2 \end{bmatrix},$$

$$V_x(\Theta, \dot{\Theta}) = \begin{bmatrix} -(m_2 l_1 c_2 + m_2 l_2)\dot{\theta}_1^2 - m_2 l_2 \dot{\theta}_2^2 - (2m_2 l_2 + m_2 l_1 c_2 + m_1 l_1 \frac{c_2}{s_2^2})\dot{\theta}_1 \dot{\theta}_2 \\ m_2 l_1 s_2 \dot{\theta}_1^2 + l_1 m_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \end{bmatrix},$$

$$G_x(\Theta) = \begin{bmatrix} m_1 g \frac{c_1}{s_2} + m_2 g s_{12} \\ m_2 g c_{12} \end{bmatrix}. \tag{6.103}$$

When $s_2 = 0$, the manipulator is in a singular position, and some of the dynamic terms go to infinity. For example, when $\theta_2 = 0$ (arm stretched straight out), the effective Cartesian mass of the end-effector becomes infinite in the $\hat{X}_2$ direction of the link-2 tip frame, as expected. In general, at a singular configuration there is a certain direction, the *singular direction* in which motion is impossible, but general motion in the subspace "orthogonal" to this direction is possible [8].

The Cartesian configuration space torque equation

Combining (6.91) and (6.92), we can write equivalent joint torques with the dynamics expressed in Cartesian space:

$$\tau = J^T(\Theta)(M_x(\Theta)\ddot{\chi} + V_x(\Theta, \dot{\Theta}) + G_x(\Theta)). \tag{6.104}$$

We will find it useful to write this equation in the form

$$\tau = J^T(\Theta)M_x(\Theta)\ddot{\chi} + B_x(\Theta)[\dot{\Theta}\dot{\Theta}] + C_x(\Theta)[\dot{\Theta}^2] + G(\Theta), \tag{6.105}$$

where $B_x(\Theta)$ is a matrix of dimension $n \times n(n-1)/2$ of Coriolis coefficients, $[\dot{\Theta}\dot{\Theta}]$ is an $n(n-1)/2 \times 1$ vector of joint velocity products given by

$$[\dot{\Theta}\dot{\Theta}] = [\dot{\theta}_1\dot{\theta}_2 \;\; \dot{\theta}_1\dot{\theta}_3 \;\; \ldots \dot{\theta}_{n-1}\dot{\theta}_n]^T, \tag{6.106}$$

$C_x(\Theta)$ is an $n \times n$ matrix of centrifugal coefficients, and $[\dot{\Theta}^2]$ is an $n \times 1$ vector given by

$$[\dot{\theta}_1^2 \;\; \dot{\theta}_2^2 \;\; \ldots \dot{\theta}_n^2]^T. \tag{6.107}$$

Note that, in (6.105), $G(\Theta)$ is the same as in the joint-space equation, but in general, $B_x(\Theta) \neq B(\Theta)$ and $C_x(\Theta) \neq C(\Theta)$.

EXAMPLE 6.7

Find $B_x(\Theta)$ and $C_x(\Theta)$ (from (6.105)) for the manipulator of Section 6.7.

If we form the product $J^T(\Theta)V_x(\Theta, \dot{\Theta})$, we find that

$$B_x(\Theta) = \begin{bmatrix} m_1 l_1^2 \frac{c_2}{s_2} - m_2 l_1 l_2 s_2 \\ m_2 l_1 l_2 s_2 \end{bmatrix} \tag{6.108}$$

and

$$C_x(\Theta) = \begin{bmatrix} 0 & -m_2 l_1 l_2 s_2 \\ m_2 l_1 l_2 s_2 & 0 \end{bmatrix}. \tag{6.109}$$

6.11 INCLUSION OF NONRIGID BODY EFFECTS

It is important to realize that the dynamic equations we have derived *do not encompass all the effects acting on a manipulator.* They include only those forces which arise from rigid body mechanics. The most important source of forces that are *not* included is friction. All mechanisms are, of course, affected by frictional forces. In present-day manipulators, in which significant gearing is typical, the forces due to friction can actually be quite large—perhaps equaling 25% of the torque required to move the manipulator in typical situations.

In order to make dynamic equations reflect the reality of the physical device, it is important to model (at least approximately) these forces of friction. A very simple model for friction is **viscous friction**, in which the torque due to friction is proportional to the velocity of joint motion. Thus, we have

$$\tau_{friction} = v\dot{\theta}, \tag{6.110}$$

where v is a viscous-friction constant. Another possible simple model for friction, **Coulomb friction**, is sometimes used. Coulomb friction is constant except for a sign dependence on the joint velocity and is given by

$$\tau_{friction} = c\, sgn(\dot{\theta}), \tag{6.111}$$

where c is a Coulomb-friction constant. The value of c is often taken at one value when $\dot{\theta} = 0$, the static coefficient, but at a lower value, the dynamic coefficient, when $\dot{\theta} \neq 0$. Whether a joint of a particular manipulator exhibits viscous or Coulomb

friction is a complicated issue of lubrication and other effects. A reasonable model is to include both, because both effects are likely:

$$\tau_{friction} = c\, sgn(\dot{\theta}) + v\dot{\theta}. \tag{6.112}$$

It turns out that, in many manipulator joints, friction also displays a dependence on the joint position. A major cause of this effect might be gears that are not perfectly round—their eccentricity would cause friction to change according to joint position. So a fairly complex friction model would have the form

$$\tau_{friction} = f(\theta, \dot{\theta}). \tag{6.113}$$

These friction models are then added to the other dynamic terms derived from the rigid-body model, yielding the more complete model

$$\tau = M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta) + F(\Theta, \dot{\Theta}). \tag{6.114}$$

There are also other effects, which are neglected in this model. For example, the assumption of rigid body links means that we have failed to include bending effects (which give rise to resonances) in our equations of motion. However, these effects are extremely difficult to model and are beyond the scope of this book. (See [9, 10].)

6.12 DYNAMIC SIMULATION

To simulate the motion of a manipulator, we must make use of a model of the dynamics such as the one we have just developed. Given the dynamics written in closed form as in (6.59), simulation requires solving the dynamic equation for acceleration:

$$\ddot{\Theta} = M^{-1}(\Theta)[\tau - V(\Theta, \dot{\Theta}) - G(\Theta) - F(\Theta, \dot{\Theta})]. \tag{6.115}$$

We can then apply any of several known **numerical integration** techniques to integrate the acceleration to compute future positions and velocities.

Given initial conditions on the motion of the manipulator, usually in the form

$$\begin{aligned} \Theta(0) &= \Theta_0, \\ \dot{\Theta}(0) &= 0, \end{aligned} \tag{6.116}$$

we integrate (6.115) forward in time numerically by steps of size Δt. There are many methods of performing numerical integration [11]. Here, we introduce the simplest integration scheme, called *Euler integration*: Starting with $t = 0$, iteratively compute

$$\begin{aligned} \dot{\Theta}(t + \Delta t) &= \dot{\Theta}(t) + \ddot{\Theta}(t)\Delta t, \\ \Theta(t + \Delta t) &= \Theta(t) + \dot{\Theta}(t)\Delta t + \tfrac{1}{2}\ddot{\Theta}(t)\Delta t^2, \end{aligned} \tag{6.117}$$

where, for each iteration, (6.115) is computed to calculate $\ddot{\Theta}$. In this way, the position, velocity, and acceleration of the manipulator caused by a certain input torque function can be computed numerically.

Euler integration is conceptually simple, but other, more sophisticated integration techniques are recommended for accurate and efficient simulation [11]. How

to select the size of Δt is an issue that is often discussed. It should be sufficiently small that breaking continuous time into these small increments is a reasonable approximation. It should be sufficiently large that an excessive amount of computer time is not required to compute a simulation.

6.13 COMPUTATIONAL CONSIDERATIONS

Because the dynamic equations of motion for typical manipulators are so complex, it is important to consider computational issues. In this section, we restrict our attention to joint-space dynamics. Some issues of computational efficiency of Cartesian dynamics are discussed in [7, 8].

A historical note concerning efficiency

Counting the number of multiplications and additions for the equations (6.46)–(6.53) when taking into consideration the simple first outward computation and simple last inward computation, we get

$$126n - 99 \text{ multiplications},$$

$$106n - 92 \text{ additions},$$

where n is the number of links (here, at least two). Although still somewhat complex, the formulation is tremendously efficient in comparison with some previously suggested formulations of manipulator dynamics. The first formulation of the dynamics for a manipulator [12, 13] was done via a fairly straightforward Lagrangian approach whose required computations came out to be approximately [14]

$$32n^4 + 86n^3 + 171n^2 + 53n - 128 \text{ multiplications},$$

$$25n^4 + 66n^3 + 129n^2 + 42n - 96 \text{ additions}.$$

For a typical case, $n = 6$, the iterative Newton–Euler scheme is about 100 times more efficient! The two approaches must of course yield equivalent equations, and numeric calculations would yield exactly the same results, but the structure of the equations is quite different. This is not to say that a Lagrangian approach cannot be made to produce efficient equations. Rather, this comparison indicates that, in formulating a computational scheme for this problem, care must be taken as regards efficiency. The relative efficiency of the method we have presented stems from posing the computations as iterations from link to link and in the particulars of how the various quantities are represented [15].

Renaud [16] and Liegois et al. [17] made early contributions concerning the formulation of the mass-distribution descriptions of the links. While studying the modeling of human limbs, Stepanenko and Vukobratovic [18] began investigating a "Newton–Euler" approach to dynamics instead of the somewhat more traditional Lagrangian approach. This work was revised for efficiency by Orin et al. [19] in an application to the legs of walking robots. Orin's group improved the efficiency somewhat by writing the forces and moments in the local link-reference frames instead of in the inertial frame. They also noticed the sequential nature of calculations from one link to the next and speculated that an efficient recursive formulation might exist. Armstrong [20] and Luh, Walker, and Paul

[2] paid close attention to details of efficiency and published an algorithm that is $O(n)$ in complexity. This was accomplished by setting up the calculations in an iterative (or recursive) nature and by expressing the velocities and accelerations of the links in the local link frames. Hollerbach [14] and Silver [15] further explored various computational algorithms. Hollerbach and Sahar [21] showed that, for certain specialized geometries, the complexity of the algorithm would reduce further.

Efficiency of closed form vs. that of iterative form

The iterative scheme introduced in this chapter is quite efficient as a general means of computing the dynamics of any manipulator, but closed-form equations derived for a particular manipulator will usually be even more efficient. Consider the two-link planar manipulator of Section 6.7. Plugging $n = 2$ into the formulas given in Section 6.13, we find that our iterative scheme would require 153 multiplications and 120 additions to compute the dynamics of a general two-link. However, our particular two-link arm happens to be quite simple: It is planar, and the masses are treated as point masses. So, if we consider the closed-form equations that we worked out in Section 6.7, we see that computation of the dynamics in this form requires about 30 multiplications and 13 additions. This is an extreme case, because the particular manipulator is very simple, but it illustrates the point that symbolic closed-form equations are likely to be the most efficient formulation of dynamics. Several authors have published articles showing that, for any given manipulator, customized closed-form dynamics are more efficient than even the best of the general schemes [22–27].

Hence, if manipulators are designed to be *simple* in the kinematic and dynamic sense, they will have dynamic equations that are simple. We might define a **kinematically simple** manipulator to be a manipulator that has many (or all) of its link twists equal to 0°, 90°, or −90° and many of its link lengths and offsets equal to zero. We might define a **dynamically simple** manipulator as one for which each link-inertia tensor is diagonal in frame $\{C_i\}$.

The drawback of formulating closed-form equations is simply that it currently requires a fair amount of human effort. However, symbolic manipulation programs that can derive the closed-form equations of motion of a device and automatically factor out common terms and perform trigonometric substitutions have been developed [25, 28–30].

Efficient dynamics for simulation

When dynamics are to be computed for the purpose of performing a numerical simulation of a manipulator, we are interested in solving for the joint accelerations, given the manipulator's current position and velocity and the input torques. An efficient computational scheme must therefore address both the computation of the dynamic equations studied in this chapter and efficient schemes for solving equations (for joint accelerations) and performing numerical integration. Several efficient methods for dynamic simulation of manipulators are reported in [31].

Memorization schemes

In any computational scheme, a trade-off can be made between computations and memory usage. In the problem of computing the dynamic equation of a manipulator (6.59), we have implicitly assumed that, when a value of τ is needed, it is computed as quickly as possible from Θ, $\dot{\Theta}$, and $\ddot{\Theta}$ at run time. If we wish, we can trade off this computational burden at the cost of a tremendously large memory by precomputing (6.59) for all possible Θ, $\dot{\Theta}$, and $\ddot{\Theta}$ values (suitably quantized). Then, when dynamic information is needed, the answer is found by table lookup.

The size of the memory required is large. Assume that each joint angle range is quantized to ten discrete values; likewise, assume that velocities and accelerations are quantized to ten ranges each. For a six-jointed manipulator, the number of cells in the $(\Theta, \dot{\Theta}, \ddot{\Theta})$ quantized space is $(10 \times 10 \times 10)^6$. In each of these cells, there are six torque values. Assuming each torque value requires one computer word, this memory size is 6×10^{18} words! Also, note that the table needs to be recomputed for a change in the mass of the load—or else another dimension needs to be added to account for all possible loads.

There are many intermediate solutions that trade off memory for computation in various ways. For example, if the matrices appearing in equation (6.63) were precomputed, the table would have only one dimension (in Θ) rather than three. After the functions of Θ are looked up, a modest amount of computation (given by (6.63)) is done. For more details and for other possible parameterizations of this problem, see [3] and [6].

BIBLIOGRAPHY

[1] I. Shames, *Engineering Mechanics*, 2nd edition, Prentice-Hall, Englewood Cliffs, NJ, 1967.

[2] J.Y.S. Luh, M.W. Walker, and R.P. Paul, "On-Line Computational Scheme for Mechanical Manipulators," *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 1980.

[3] M. Raibert, "Mechanical Arm Control Using a State Space Memory," SME paper MS77-750, 1977.

[4] K.R. Symon, *Mechanics*, 3rd edition, Addison-Wesley, Reading, MA, 1971.

[5] B. Noble, *Applied Linear Algebra*, Prentice-Hall, Englewood Cliffs, NJ, 1969.

[6] O. Khatib, "Commande Dynamique dans L'Espace Operationnel des Robots Manipulateurs en Presence d'Obstacles," These de Docteur-Ingenieur. Ecole Nationale Superieure de l'Aeronautique et de L'Espace (ENSAE), Toulouse.

[7] O. Khatib, "Dynamic Control of Manipulators in Operational Space," Sixth IFTOMM Congress on Theory of Machines and Mechanisms, New Delhi, December 15–20, 1983.

[8] O. Khatib, "The Operational Space Formulation in Robot Manipulator Control," 15th ISIR, Tokyo, September 11–13, 1985.

[9] E. Schmitz, "Experiments on the End-Point Position Control of a Very Flexible One-Link Manipulator," Unpublished Ph.D. Thesis, Department of Aeronautics and Astronautics, Stanford University, SUDAAR No. 547, June 1985.

[10] W. Book, "Recursive Lagrangian Dynamics of Flexible Manipulator Arms," *International Journal of Robotics Research*, Vol. 3, No. 3, 1984.

[11] S. Conte and C. DeBoor, *Elementary Numerical Analysis: An Algorithmic Approach*, 2nd edition, McGraw-Hill, New York, 1972.

[12] J. Uicker, "On the Dynamic Analysis of Spatial Linkages Using 4 x 4 Matrices," Unpublished Ph.D dissertation, Northwestern University, Evanston, IL, 1965.

[13] J. Uicker, "Dynamic Behaviour of Spatial Linkages," *ASME Mechanisms*, Vol. 5, No. 68, pp. 1–15.

[14] J.M. Hollerbach, "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity," in *Robot Motion*, M. Brady et al., Editors, MIT Press, Cambridge, MA, 1983.

[15] W. Silver, "On the Equivalence of Lagrangian and Newton–Euler Dynamics for Manipulators," *International Journal of Robotics Research*, Vol. 1, No. 2, pp. 60–70.

[16] M. Renaud, "Contribution à l'Etude de la Modélisation et de la Commande des Systèmes Mécaniques Articulés," Thèse de Docteur-Ingénieur, Université Paul Sabatier, Toulouse, December 1975.

[17] A. Liegois, W. Khalil, J.M. Dumas, and M. Renaud, "Mathematical Models of Interconnected Mechanical Systems," Symposium on the Theory and Practice of Robots and Manipulators, Poland, 1976.

[18] Y. Stepanenko and M. Vukobratovic, "Dynamics of Articulated Open-Chain Active Mechanisms," *Math-Biosciences* Vol. 28, 1976, pp. 137–170.

[19] D.E. Orin et al, "Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton–Euler Methods," *Math-Biosciences* Vol. 43, 1979, pp. 107–130.

[20] W.W. Armstrong, "Recursive Solution to the Equations of Motion of an N-Link Manipulator," *Proceedings of the 5th World Congress on the Theory of Machines and Mechanisms*, Montreal, July 1979.

[21] J.M. Hollerbach and G. Sahar, "Wrist-Partitioned Inverse Accelerations and Manipulator Dynamics," MIT AI Memo No. 717, April 1983.

[22] T.K. Kanade, P.K. Khosla, and N. Tanaka, "Real-Time Control of the CMU Direct Drive Arm II Using Customized Inverse Dynamics," *Proceedings of the 23rd IEEE Conference on Decision and Control*, Las Vegas, NV, December 1984.

[23] A. Izaguirre and R.P. Paul, "Computation of the Inertial and Gravitational Coefficients of the Dynamic Equations for a Robot Manipulator with a Load," *Proceedings of the 1985 International Conference on Robotics and Automation*, pp. 1024–1032, St. Louis, March 1985.

[24] B. Armstrong, O. Khatib, and J. Burdick, "The Explicit Dynamic Model and Inertial Parameters of the PUMA 560 Arm," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, April 1986, pp. 510–518.

[25] J.W. Burdick, "An Algorithm for Generation of Efficient Manipulator Dynamic Equations," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, April 7–11, 1986, pp. 212–218.

[26] T.R. Kane and D.A. Levinson, "The Use of Kane's Dynamical Equations in Robotics," *The International Journal of Robotics Research*, Vol. 2, No. 3, Fall 1983, pp. 3–20.

[27] M. Renaud, "An Efficient Iterative Analytical Procedure for Obtaining a Robot Manipulator Dynamic Model," First International Symposium of Robotics Research, NH, August 1983.

[28] W. Schiehlen, "Computer Generation of Equations of Motion," in *Computer Aided Analysis and Optimization of Mechanical System Dynamics*, E.J. Haug, Editor, Springer-Verlag, Berlin & New York, 1984.

[29] G. Cesareo, F. Nicolo, and S. Nicosia, "DYMIR: A Code for Generating Dynamic Model of Robots," in *Advanced Software in Robotics*, Elsevier Science Publishers, North-Holland, 1984.

[30] J. Murray, and C. Neuman, "ARM: An Algebraic Robot Dynamic Modelling Program," IEEE International Conference on Robotics, Atlanta, March 1984.

[31] M. Walker and D. Orin, "Efficient Dynamic Computer Simulation of Robotic Mechanisms," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 104, 1982.

EXERCISES

6.1 [12] Find the inertia tensor of a right cylinder of homogeneous density with respect to a frame with origin at the center of mass of the body.

6.2 [32] Construct the dynamic equations for the two-link manipulator in Section 6.7 when each link is modeled as a rectangular solid of homogeneous density. Each link has dimensions l_i, ω_i, and h_i and total mass m_i.

6.3 [43] Construct the dynamic equations for the three-link manipulator of Chapter 3, Exercise 3.3. Consider each link to be a rectangular solid of homogeneous density with dimensions l_i, ω_i, and h_i and total mass m_i.

6.4 [13] Write the set of equations that correspond to (6.46)–(6.53) for the case where the mechanism could have sliding joints.

6.5 [30] Construct the dynamic equations for the two-link nonplanar manipulator shown in Fig. 6.8. Assume that all the mass of the links can be considered as a point mass located at the distal (outermost) end of the link. The mass values are m_1 and m_2, and the link lengths are l_1 and l_2. This manipulator is like the first two links of the arm in Exercise 3.3. Assume further that viscous friction is acting at each joint, with coefficients v_1 and v_2.

6.6 [32] Derive the Cartesian space form of the dynamics for the two-link planar manipulator of Section 6.7 in terms of the base frame. *Hint*: See Example 6.5, but use the Jacobian written in the base frame.

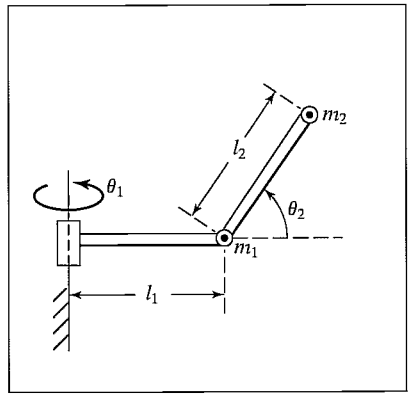

FIGURE 6.8: Two-link nonplanar manipulator with point masses at distal ends of links.

6.7 [18] How many memory locations would be required to store the dynamic equations of a general three-link manipulator in a table? Quantize each joint's position, velocity, and acceleration into 16 ranges. Make any assumptions needed.

6.8 [32] Derive the dynamic equations for the two-link manipulator shown in Fig. 4.6. Link 1 has an inertia tensor given by

$$C_1 I = \begin{bmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{bmatrix}.$$

Assume that link 2 has all its mass, m_2, located at a point at the end-effector. Assume that gravity is directed downward (opposite $\hat{Z}_1$).

6.9 [37] Derive the dynamic equations for the three-link manipulator with one prismatic joint shown in Fig. 3.9. Link 1 has an inertia tensor given by

$$C_1 I = \begin{bmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{bmatrix}.$$

Link 2 has point mass m_2 located at the origin of its link frame. Link 3 has an inertia tensor given by

$$C_3 I = \begin{bmatrix} I_{xx3} & 0 & 0 \\ 0 & I_{yy3} & 0 \\ 0 & 0 & I_{zz3} \end{bmatrix}.$$

Assume that gravity is directed opposite $\hat{Z}_1$ and that viscous friction of magnitude v_i is active at each joint.

6.10 [35] Derive the dynamic equations in Cartesian space for the manipulator of Exercise 6.8. Write the equations in frame {2}.

6.11 [20] A certain one-link manipulator has

$$C_1 I = \begin{bmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{bmatrix}.$$

Assume that this is just the inertia of the link itself. If the motor armature has a moment of inertia I_m and the gear ratio is 100, what is the total inertia as seen from the motor shaft [1]?

6.12 [20] The single-degree-of-freedom "manipulator" in Fig. 6.9 has total mass $m = 1$, with the center of mass at

$${}^1P_C = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix},$$

and has inertia tensor

$${}^C I_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}.$$

From rest at $t = 0$, the joint angle θ_1 moves in accordance with the time function

$$\theta_1(t) = bt + ct^2$$

in radians. Give the angular acceleration of the link and the linear acceleration of the center of mass in terms of frame {1} as a function of t.

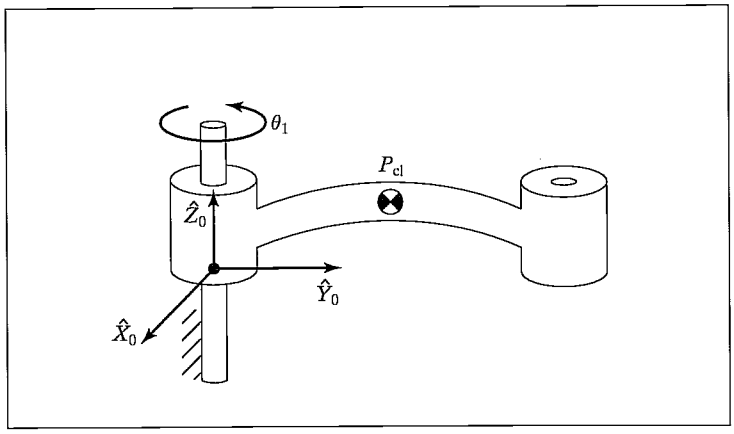

FIGURE 6.9: One-link "manipulator" of Exercise 6.12.

6.13 [40] Construct the Cartesian dynamic equations for the two-link nonplanar manipulator shown in Fig. 6.8. Assume that all the mass of the links can be considered as a point mass located at the distal (outermost) end of the link. The mass values are m_1 and m_2, and the link lengths are l_1 and l_2. This manipulator is like the first two links of the arm in Exercise 3.3. Also assume that viscous friction is acting at each joint with coefficients v_1 and v_2. Write the Cartesian dynamics in frame {3}, which is located at the tip of the manipulator and has the same orientation as link frame {2}.

6.14 [18] The following equations were derived for a 2-DOF RP manipulator:

$$\tau_1 = m_1(d_1^2 + d_2)\ddot{\theta}_1 + m_2 d_2^2\ddot{\theta}_1 + 2m_2 d_2\dot{d}_2\dot{\theta}_1 + g\cos(\theta_1)[m_1(d_1 + d_2\dot{\theta}_1) + m_2(d_2 + \dot{d}_2)]$$

$$\tau_2 = m_1\dot{d}_2\ddot{\theta}_1 + m_2\ddot{d}_2 - m_1 d_1\dot{d}_2 - m_2 d_2\dot{\theta}^2 + m_2(d_2 + 1)g\sin(\theta_1).$$

Some of the terms are obviously incorrect. Indicate the incorrect terms.

6.15 [28] Derive the dynamic equations for the RP manipulator of Example 6.5, using the Newton–Euler procedure instead of the Lagrangian technique.

6.16 [25] Derive the equations of motion for the PR manipulator shown in Fig. 6.10. Neglect friction, but include gravity. (Here, $\hat{X}_0$ is upward.) The inertia tensors of the links are diagonal, with moments $I_{xx1}, I_{yy1}, I_{zz1}$ and $I_{xx2}, I_{yy2}, I_{zz2}$. The centers of mass for the links are given by

$${}^1P_{C_1} = \begin{bmatrix} 0 \\ 0 \\ -l_1 \end{bmatrix},$$

$${}^2P_{C_2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

6.17 [40] The velocity-related terms appearing in the manipulator dynamic equation can be written as a matrix-vector product—that is,

$$V(\Theta, \dot{\Theta}) = V_m(\Theta, \dot{\Theta})\dot{\Theta},$$

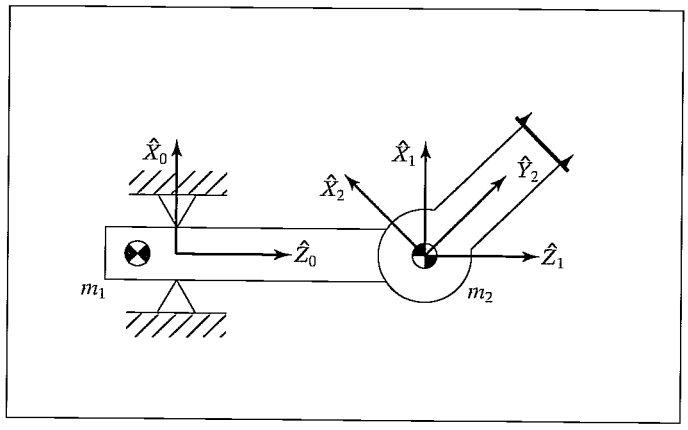

FIGURE 6.10: PR manipulator of Exercise 6.16.

where the m subscript stands for "matrix form." Show that an interesting relationship exists between the time derivative of the manipulator mass matrix and $V_m(\cdot)$, namely,

$$\dot{M}(\Theta) = 2V_m(\Theta, \dot{\Theta}) - S,$$

where S is some skew-symmetric matrix.

6.18 [15] Give two properties that any reasonable friction model (i.e., the term $F(\Theta, \dot{\Theta})$ in (6.114)) would possess.

6.19 [28] Do Exercise 6.5, using Lagrange's equations.

6.20 [28] Derive the dynamic equations of the 2-DOF manipulator of Section 6.7, using a Lagrangian formulation.

PROGRAMMING EXERCISE (PART 6)

1. Derive the dynamic equations of motion for the three-link manipulator (from Example 3.3). That is, expand Section 6.7 for the three-link case. The following numerical values describe the manipulator:

$$\begin{aligned} l_1 &= l_2 = 0.5\text{m}, \\ m_1 &= 4.6\text{Kg}, \\ m_2 &= 2.3\text{Kg}, \\ m_3 &= 1.0\text{Kg}, \\ g &= 9.8\text{m/s}^2. \end{aligned}$$

For the first two links, we assume that the mass is all concentrated at the distal end of the link. For link 3, we assume that the center of mass is located at the origin of frame {3}—that is, at the proximal end of the link. The inertia tensor for link 3 is

$${}^{C_3}I = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \text{Kg-m}^2.$$

The vectors that locate each center of mass relative to the respective link frame are

$$^1P_{C_1} = l_1\hat{X}_1,$$
$$^2P_{C_2} = l_2\hat{X}_2,$$
$$^3P_{C_3} = 0.$$

2. Write a simulator for the three-link manipulator. A simple Euler-integration routine is sufficient for performing the numerical integration (as in Section 6.12). To keep your code modular, it might be helpful to define the routine

```
Procedure UPDATE(VAR tau: vec3; VAR period: real; VAR
theta, thetadot: vec3);
```

where "tau" is the torque command to the manipulator (always zero for this assignment), "period" is the length of time you wish to advance time (in seconds), and "theta" and "thetadot" are the *state* of the manipulator. Theta and thetadot are updated by "period" seconds each time you call UPDATE. Note that "period" would typically be longer than the integration step size, Δt, used in the numerical integration. For example, although the step size for numerical integration might be 0.001 second, you might wish to print out the manipulator position and velocity only each 0.1 seconds.

To test your simulation, set the joint-torque commands to zero (for all time) and perform these tests:

(a) Set the initial position of the manipulator to

$$[\theta_1\ \theta_2\ \theta_3] = [-90\ 0\ 0].$$

Simulate for a few seconds. Is the motion of the manipulator what you would expect?

(b) Set the initial position of the manipulator to

$$[\theta_1\ \theta_2\ \theta_3] = [30\ 30\ 10].$$

Simulate for a few seconds. Is the motion of the manipulator what you would expect?

(c) Introduce some viscous friction at each joint of the simulated manipulator—that is, add a term to the dynamics of each joint in the form $\tau_f = v\dot{\theta}$, where $v = 5.0$ newton-meter-seconds for each joint. Repeat test (b) above. Is the motion what you would expect?

MATLAB EXERCISE 6A

This exercise focuses on the inverse-dynamics analysis (in a resolved-rate control framework—see MATLAB Exercise 5) for the planar 2-DOF 2R robot. This robot is the first two R-joints and first two moving links of the planar 3-DOF 3R robot. (See Figures 3.6 and 3.7; the DH parameters are given in the first two rows of Figure 3.8.)

For the planar 2R robot, calculate the required joint torques (i.e., solve the inverse-dynamics problem) to provide the commanded motion at every time step in a resolved-rate control scheme. You can use either numerical Newton–Euler recursion or the analytical equations from the results of Exercise 6.2, or both.

Given: $L_1 = 1.0$ m, $L_2 = 0.5$ m; Both links are solid steel with mass density $\rho = 7806$ kg/m^3; both have the width and thickness dimensions $w = t = 5$ cm. The revolute joints are assumed to be perfect, connecting the links at their very edges (not physically possible).

The initial angles are $\Theta = \begin{Bmatrix} \theta_1 \\ \theta_2 \end{Bmatrix} = \begin{Bmatrix} 10^\circ \\ 90^\circ \end{Bmatrix}$.

The (constant) commanded Cartesian velocity is ${}^0\dot{X} = {}^0\begin{Bmatrix} \dot{x} \\ \dot{y} \end{Bmatrix} = {}^0\begin{Bmatrix} 0 \\ 0.5 \end{Bmatrix}$ (m/s).

Simulate motion for 1 sec, with a control time step of 0.01 sec.
Present five plots (each set on a separate graph, please):

1. the two joint angles (degrees) $\Theta = \{\theta_1 \ \ \theta_2\}^T$ vs. time;
2. the two joint rates (rad/s) $\dot{\Theta} = \{\dot{\theta}_1 \ \ \dot{\theta}_2\}^T$ vs. time;
3. the two joint accelerations (rad/s^2) $\ddot{\Theta} = \{\ddot{\theta}_1 \ \ \ddot{\theta}_2\}^T$ vs. time;
4. the three Cartesian components of ${}^0_H T$, $X = \{x \ \ y \ \ \phi\}^T$ (rad is fine for ϕ so it will fit) vs. time;
5. the two inverse dynamics joint torques (Nm) $\mathrm{T} = \{\tau_1 \ \ \tau_2\}^T$ vs. time.

Carefully label (by hand is fine!) each component on each plot. Also, label the axis names and units.

Perform this simulation twice. The first time, ignore gravity (the motion plane is normal to the effect of gravity); the second time, consider gravity g in the negative Y direction.

MATLAB EXERCISE 6B

This exercise focuses on the inverse-dynamics solution for the planar 3-DOF, 3R robot (of Figures 3.6 and 3.7; the DH parameters are given in Figure 3.8) for a motion snapshot in time only. The following fixed-length parameters are given: $L_1 = 4$, $L_2 = 3$, and $L_3 = 2$ (m). For dynamics, we must also be given mass and moment-of-inertia information: $m_1 = 20$, $m_2 = 15$, $m_3 = 10$ (kg), ${}^C I_{ZZ1} = 0.5$, ${}^C I_{ZZ2} = 0.2$, and ${}^C I_{ZZ3} = 0.1$ (kgm^2). Assume that the CG of each link is in its geometric center. Also, assume that gravity acts in the $-Y$ direction in the plane of motion. For this exercise, ignore actuator dynamics and the joint gearing.

a) Write a MATLAB program to implement the recursive Newton–Euler inverse-dynamics solution (i.e., given the commanded motion, calculate the required driving joint torques) for the following motion snapshot in time:

$$\Theta = \begin{Bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{Bmatrix} = \begin{Bmatrix} 10^\circ \\ 20^\circ \\ 30^\circ \end{Bmatrix} \quad \dot{\Theta} = \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 3 \end{Bmatrix} \text{(rad/s)} \ddot{\Theta} = \begin{Bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{Bmatrix} = \begin{Bmatrix} 0.5 \\ 1 \\ 1.5 \end{Bmatrix} \text{(rad/s}^2\text{)}$$

b) Check your results in (a) by means of the Corke MATLAB Robotics Toolbox. Try functions *rne()* and *gravload()*.

MATLAB EXERCISE 6C

This exercise focuses on the forward-dynamics solution for the planar 3-DOF, 3R robot (parameters from MATLAB Exercise 6B) for motion over time. In this case, ignore gravity (i.e., assume that gravity acts in a direction normal to the plane of motion). Use the Corke MATLAB Robotics Toolbox to solve the forward-dynamics problem (i.e.,

given the commanded driving joint torques, calculate the resulting robot motion) for the following constant joint torques and the given initial joint angles and initial joint rates:

$$\mathrm{T} = \begin{Bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{Bmatrix} = \begin{Bmatrix} 20 \\ 5 \\ 1 \end{Bmatrix} \text{ (Nm, constant)} \quad \Theta_0 = \begin{Bmatrix} \theta_{10} \\ \theta_{20} \\ \theta_{30} \end{Bmatrix} = \begin{Bmatrix} -60^\circ \\ 90^\circ \\ 30^\circ \end{Bmatrix}$$

$$\dot{\Theta}_0 = \begin{Bmatrix} \dot{\theta}_{10} \\ \dot{\theta}_{20} \\ \dot{\theta}_{30} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \text{ (rad/s)}$$

Perform this simulation for 4 seconds. Try function *fdyn()*.

Present two plots for the resulting robot motion (each set on a separate graph, please):

1. the three joint angles (degrees) $\Theta = \{\theta_1 \ \theta_2 \ \theta_3\}^T$ vs. time;
2. the three joint rates (rad/s) $\dot{\Theta} = \{\dot{\theta}_1 \ \dot{\theta}_2 \ \dot{\theta}_3\}^T$ vs. time.

Carefully label (by hand is fine!) each component on each plot. Also, label the axis names and units.