5
Dynamic Analysis. Equations of Motion

This chapter deals with the *direct dynamic problem* which consists of determining the motion of a multibody system that results from the application of the external forces and/or the kinematically controlled or driven degrees of freedom. The direct dynamic analysis is also commonly referred to as the *dynamic simulation*. Its importance is steadily increasing in fields such as: automobile industry, aerospace, robotics, machinery, biomechanics, and others. The possibility of kinematically controlling some degrees of freedom in a dynamic problem has many practical applications. For example, in the analysis of vehicle suspensions, if the wheel is rigid, its center follows the trajectory determined by the rolling surface. The dynamic problem will determine the resulting motion of all the vehicle's remaining elements.

It is very important to emphasize the difference between the kinematically and dynamically controlled degrees of freedom. In the previously stated kinematic simulation, all the degrees of freedom were controlled kinematically; that is, the motion of as many input elements as degrees of freedom is known. There are as many additional kinematic or driving constraint equations such as known angles, and known distances, as degrees of freedom. In order for the problem to be truly dynamic, it is necessary that the number of unknown dependent variables be greater than the total number of independent geometric and driving constraint equations. As a result, the motion of the multibody system cannot be unequivocally defined by the geometric and driving constraint equations and by the known motion of points and vectors only. In order to determine the motion of the entire system, it is necessary to establish the dynamic equilibrium condition that leads to a system of second order differential equations generally called the *equations of motion*.

The direct dynamic problem pursues the determination of the system's motion during a period of time originated by known external forces and/or the kinematically driven degrees of freedom. The position of the multibody system is characterized by its dependent coordinates. It is not sufficient to know the values of a minimum set of independent coordinates, because the position is not unequivocally known by simply knowing the independent coordinates. However, at the time of formulating the equations of motion, it is possible to do it with both dependent or independent coordinates. There is not a consensus among the ex-

perts as to which method is the best for all cases. A method can be advantageous over another under certain conditions and vice versa. Continuing research is being carried out to find the best possible formulation in terms of efficiency and accuracy.

The dynamic formulation with independent coordinates calls for solving the position problem at each stage of the integration or adopting alternative methods that will be later explained below. In practice this is not a serious problem. As the positions of the system corresponding to two consecutive steps of the integration are very close, the position problem converges rapidly and the problem of multiple solutions does not present a serious practical difficulty.

We discuss in this chapter several methods concerning formulating and solving the direct dynamic problem with both dependent (Section 5.1) and independent coordinates (Section 5.2). In many instances we include algorithms which explain and facilitate their computer implementation. Some recent formulations that are based on velocity transformations and the canonical equations are discussed in Sections 5.3 and 5.4, respectively.

5.1 Formulations in Dependent Coordinates

Several methods of formulating the equations of motion with dependent coordinates will be developed below. In all cases, the desired end result can be obtained by either the Lagrange's equations or the method of virtual power. Hereafter, the vector $\mathbf{q}$ represents a set of n unknown dependent coordinates, m is the total number of independent constraint equations (geometric and kinematic), and $f=n-m$ is the number of dynamic degrees of freedom. The constraint conditions are written in the following general form:

$$\boldsymbol{\Phi}(\mathbf{q}, t) = \mathbf{0} \tag{5.1}$$

Let $T(\mathbf{q}, \dot{\mathbf{q}})$ be the kinetic energy of the system, $V(\mathbf{q})$ the potential energy and $\mathbf{Q}_{ex}(\mathbf{q})$ the vector of generalized external forces acting along the dependent coordinates $\mathbf{q}$ of a constrained mechanical system. The Lagrange's equations of such systems have been derived in Chapter 4 in the form:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial L}{\partial \mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} \boldsymbol{\lambda} = \mathbf{Q}_{ex} \tag{5.2}$$

where $L=T-V$ is the *Lagrangian* function. The third term on the LHS of equation (5.2) is introduced, because the coordinates $\mathbf{q}$ are not independent but interrelated by means of the constraint equations. The matrix $\boldsymbol{\Phi}_{\mathbf{q}}$ is the Jacobian matrix of the nonlinear constraint equations (5.1). The vector $\boldsymbol{\lambda}$ in (5.2) represents the Lagrange multipliers. With this formulation the number of unknowns has increased to $n+m$, since not only $\mathbf{q}$ but also $\boldsymbol{\lambda}$ needs to be calculated.

As shown in Chapter 4, the kinetic energy of a multibody system can be written as follows:

$$T = \frac{1}{2}\,\dot{\mathbf{q}}^{\mathrm{T}}\,\mathbf{M}(\mathbf{q})\,\dot{\mathbf{q}} \tag{5.3}$$

where the mass matrix is constant as long as all the bodies have at least two points and two non-coplanar unit vectors or an equivalent structure. Otherwise, the mass matrix is dependent on the positions **q**. For the general case in which the kinetic energy depends on **q**, equation (5.2) becomes (See Example 4.1)

$$\mathbf{M}\,\ddot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda} = \mathbf{Q}_{\mathrm{ex}} + L_{\mathbf{q}} - \dot{\mathbf{M}}\,\dot{\mathbf{q}} \tag{5.4}$$

where $\mathbf{Q}_{\mathrm{ex}}$ is the vector of external forces and L the Lagrangian. It may be seen that equation (5.2) involves the time differentiation of the mass matrix which leads in certain cases to rather involved computations.

Another way of formulating the equations of motion is by means of the *method of virtual power*. It was demonstrated in Chapter 4 that the virtual power of the forces acting on a multibody system can be written as

$$\dot{\mathbf{q}}^{*\mathrm{T}}\left(\mathbf{M}\,\ddot{\mathbf{q}} - \mathbf{Q}\right) = 0 \tag{5.5}$$

where $\dot{\mathbf{q}}^{*}$ are the virtual velocities, which must satisfy the first derivative of the constraint equations at a stationary time. Therefore,

$$\boldsymbol{\Phi}_{\mathbf{q}}\left(\mathbf{q}, t\right)\dot{\mathbf{q}}^{*} = \mathbf{0} \tag{5.6}$$

It cannot be inferred from equation (5.5), that the part of the expression between parenthesis is zero. In addition to the inertia and external forces coming from a potential, the constraint forces, such as forces at the pairs, also act on the multibody system. Although they do not appear in a virtual power expression, they should appear in the equilibrium equations. In order to eliminate the virtual dependent velocities from equation (5.5), one should add to (5.5) the equation (5.6) transposed and multiplied by a vector of m unknown coefficients $\boldsymbol{\lambda}$. This would yield:

$$\dot{\mathbf{q}}^{*\mathrm{T}}\left(\mathbf{M}\,\ddot{\mathbf{q}} - \mathbf{Q} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda}\right) = 0 \tag{5.7}$$

As mentioned in Section 4.1, it is always possible to find m values of the vector $\boldsymbol{\lambda}$. This vector establishes the magnitude of the constraint forces; so that

$$\mathbf{M}\,\ddot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda} = \mathbf{Q} \tag{5.8}$$

where now the vector **Q** contains the external forces plus all the velocity-dependent inertia terms obtained as explained in Chapter 4. Hence, equations (5.4) and (5.8) are equivalent. The first term corresponds to the inertia forces; the last, to the external forces, velocity-dependent inertia forces, and those obtained from a potential. The intermediate term corresponds to the forces associated with the imposed constraints, that is, the forces necessary for the constraint equations to be satisfied.

5.1.1 Method of the Lagrange Multipliers

Equation (5.8) represents n equations and $(n+m)$ unknowns: the n elements of vector $\ddot{\mathbf{q}}$ and the m elements of vector $\boldsymbol{\lambda}$. In order to have a sufficient number of equations, it is necessary to supply m more equations. The obvious choice is to use the algebraic constraint equations (5.1) which along with (5.8) constitute a set of differential algebraic equations DAEs of index three (See Chapter 7). In order to avoid DAEs, one can use the acceleration kinematic equations which are obtained by differentiating the constraint equations (5.1) twice with respect to time:

$$\boldsymbol{\Phi}_{\mathbf{q}}\,\ddot{\mathbf{q}} = -\dot{\boldsymbol{\Phi}}_t - \dot{\boldsymbol{\Phi}}_{\mathbf{q}}\,\dot{\mathbf{q}} \equiv \mathbf{c} \tag{5.9}$$

This expression is used to define the vector $\mathbf{c}$. By writing equations (5.8) and (5.9) jointly, one obtains:

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ \mathbf{c} \end{Bmatrix} \tag{5.10}$$

which is a system of $(n+m)$ equations with $(n+m)$ unknowns, whose matrix is symmetrical and, in general, non-positive definite, and also very sparse in many practical cases.

The system of equations (5.10) can be used for the simultaneous solution of the accelerations and Lagrange multipliers. Alternatively, equation (5.8) can be solved first to obtain an expression for the accelerations:

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}\,\mathbf{Q} - \mathbf{M}^{-1}\,\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda} \tag{5.11}$$

which can only be used if the mass matrix is non-singular, as it will be in most of the cases. By substituting equation (5.11) in equation (5.9), one obtains:

$$\boldsymbol{\Phi}_{\mathbf{q}}\,\mathbf{M}^{-1}\,\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda} = \boldsymbol{\Phi}_{\mathbf{q}}\,\mathbf{M}^{-1}\,\mathbf{Q} - \mathbf{c} \tag{5.12}$$

from which the Lagrange multiplier vector $\boldsymbol{\lambda}$ can be found. In order to calculate the accelerations, it will suffice to substitute $\boldsymbol{\lambda}$ in equation (5.11).

In the majority of practical cases, the direct solution of equations (5.10) is preferable over the use of (5.11) and (5.12). The main advantage of the dynamic formulation in dependent coordinates using Lagrange multipliers, besides the conceptual simplicity of the method, is permitting the calculation of forces associated with the constraints (which depend on the Lagrange multipliers) with a minimum additional effort. The solution of (5.10) yields $\boldsymbol{\lambda}$ directly without the need for a special call to an inverse dynamic module (See Chapter 6).

Numerical Implementation. It will be assumed that a numerical integration subroutine for first order differential equations, such as those described in Chapter 7, is available. The operation of these subroutines can be summarized as follows: given the vector of derivatives $\dot{\mathbf{y}}_t$ of the dependent variables at time t, the numeri-

cal integration subroutine (n.i.s.) returns the value of the vector $\mathbf{y}$ at time $(t+\Delta t)$. Schematically,

$$\dot{\mathbf{y}}_t \xrightarrow{\text{n.i.s.}} \mathbf{y}_{t+\Delta t}$$

Therefore, following the Lagrange multiplier method, the numerical integration of the equations of motion may proceed as follows:

Algorithm 5-1

1. Start at a time t in which the position $\mathbf{q}$ and velocity $\dot{\mathbf{q}}$ are known.
2. Use equations (5.10) to solve the accelerations at time t . We call this process a *function evaluation*.
3. The vector $\dot{\mathbf{y}}_t^T \equiv \{\ddot{\mathbf{q}}^T, \dot{\mathbf{q}}^T\}_t$ is given as input to the numerical integration subroutine (valid for first order differential equations), and the vector $\mathbf{y}_{t+\Delta t}^T \equiv \{\dot{\mathbf{q}}^T, \mathbf{q}^T\}_{t+\Delta t}$ is obtained:

$$\dot{\mathbf{y}}_t^T \equiv \{\ddot{\mathbf{q}}^T, \dot{\mathbf{q}}^T\}_t \xrightarrow{\text{n.i.s.}} \mathbf{y}_{t+\Delta t}^T \equiv \{\dot{\mathbf{q}}^T, \mathbf{q}^T\}_{t+\Delta t}$$

4. Upon convergence of the n.i.s., update the time variable and go to step 2.

This numerical integration algorithm has the advantage of being much simpler than those shown below corresponding to other methods. However, it may not be the most efficient. In addition, it will be explained below that as the numerical integration proceeds using this algorithm, the constraint conditions are progressively violated leading to unacceptable results in all but very short simulations.

5.1.2 Method Based on the Projection Matrix **R**

A second possibility of formulating the motion differential equations with dependent coordinates is based on the matrix $\mathbf{R}$ introduced in Chapter 3. Remember that the $f=n-m$ columns of the matrix $\mathbf{R}$ represent a basis of the nullspace of the Jacobian $\mathbf{\Phi_q}$; that is, a basis of the subspace of possible motions. The matrix $\mathbf{R}$ verifies the following relationship for holonomic systems:

$$\mathbf{\Phi_q}\,\mathbf{R} = \mathbf{0} \tag{5.13}$$

It also directly relates the dependent and independent velocities for the case in which there are no rheonomous constraints:

$$\dot{\mathbf{q}} = \mathbf{R}\,\dot{\mathbf{z}} \tag{5.14}$$

If in the virtual power equation (5.5) the dependent virtual velocities $\dot{\mathbf{q}}^*$ are substituted in terms of the independent virtual velocities $\dot{\mathbf{z}}^*$, the use of equation (5.14) leads to

$$\dot{\mathbf{z}}^{*T}\,\mathbf{R}^T\left(\mathbf{M}\,\ddot{\mathbf{q}} - \mathbf{Q}\right) = \mathbf{0} \tag{5.15}$$

Since the previous expression should be verified for any arbitrary vector of independent virtual velocities, the following must also be satisfied:

$$\mathbf{R}^T \mathbf{M}\, \ddot{\mathbf{q}} = \mathbf{R}^T \mathbf{Q} \tag{5.16}$$

Equation (5.16) contains (n–m) equations with n unknowns. In order to have as many equations as unknowns, it is necessary to complete this system with the kinematic acceleration equations (5.9), resulting in

$$\begin{bmatrix} \boldsymbol{\Phi}_\mathbf{q} \\ \mathbf{R}^T \mathbf{M} \end{bmatrix} \ddot{\mathbf{q}} = \begin{Bmatrix} \mathbf{c} \\ \mathbf{R}^T \mathbf{Q} \end{Bmatrix} \tag{5.17}$$

which is a system of n equations with n unknowns which can be easily solved for the dependent accelerations $\ddot{\mathbf{q}}$. The upper part of equation (5.17), corresponding to matrix $\boldsymbol{\Phi}_\mathbf{q}$, has been previously factored in order to calculate the matrix $\mathbf{R}$. Because of this, the system of equations can be solved with very little additional effort. The method based on equation (5.17) is sometimes more efficient than the one based on equations (5.10) (Unda et al. (1987)). The dynamic formulation whose end result is equation (5.17) was originally introduced by Kamman and Huston (1984), although they did not use a general matrix $\mathbf{R}$ but a set of eigenvectors associated with the zero eigenvalues of the matrix ($\boldsymbol{\Phi}_\mathbf{q}^T \boldsymbol{\Phi}_\mathbf{q}$).

Matrix $\mathbf{R}$ can be calculated by means of any of the methods explained in Chapter 3. The simplest is the projection method (Section 5.2.3) based on the selection of the independent coordinates as a subset of the dependent ones.

Remarks:

* The same result of equation (5.16) can be arrived at by eliminating the vector $\boldsymbol{\lambda}$ in equation (5.8). By multiplying equation (5.8) by the matrix $\mathbf{R}^T$, we can write

$$\mathbf{R}^T \mathbf{M}\, \ddot{\mathbf{q}} + \mathbf{R}^T \boldsymbol{\Phi}_\mathbf{q}^T \boldsymbol{\lambda} = \mathbf{R}^T \mathbf{Q} \tag{5.18}$$

  but by virtue of equation (5.13), the term containing the Lagrange multipliers can be cancelled, thus equation (5.16) is obtained.

* Equation (5.17) allows one to clearly distinguish the equations corresponding to the kinematics (the m first ones) from the equations corresponding to the dynamics (the n–m last ones). Besides, system (5.17) does not explicitly contain any independent coordinates; rather they are implicitly considered via the matrix $\mathbf{R}$. Each matrix $\mathbf{R}$ implies a choice of independent coordinates in accordance with equation (5.14). The chosen set of independent coordinates must be changed anytime there is a need to guarantee the existence and perfect conditioning of the Jacobian factorization necessary to compute the matrix $\mathbf{R}$.

Numerical Implementation. Similar to the case of the Lagrange multiplier method, the numerical integration process of the equations of motion using the matrix $\mathbf{R}$ may proceed as follows:

Algorithm 5-2

1. Start at a time t in which the position $\mathbf{q}$ and velocity $\dot{\mathbf{q}}$, are known.
2. Form the matrix $\mathbf{\Phi_q}$ and triangularize it with column partial pivoting. From this triangularization, decide whether the current set of coordinates continue to be valid (independent) to form the matrix $\mathbf{R}$, or if it is necessary to change them. In the latter case, make a new choice of independent columns by means of a triangularization with total pivoting.
3. Form the matrix $\mathbf{R}$ and the product $\mathbf{R}^T\mathbf{M}$.
4. Solve equation (5.17) for the dependent accelerations.
5. Obtain the vectors $\mathbf{q}$ and $\dot{\mathbf{q}}$ at time $(t+\Delta t)$ are obtained by numerical integration.

$$\dot{\mathbf{y}}_t^T \equiv \{\ddot{\mathbf{q}}^T, \dot{\mathbf{q}}^T\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}_{t+\Delta t}^T \equiv \{\dot{\mathbf{q}}^T, \mathbf{q}^T\}_{t+\Delta t}$$

6. Upon convergence of the n.i.s., update the time variable and go to step 2.

Similar to Algorithm 5-1, this method also requires constraint stabilization for long simulations. This point is treated next.

5.1.3 Stabilization of the Constraint Equations

Once the position, velocity, and external forces are known, equations (5.10) and (5.17) permit calculating the dependent accelerations of the system at a specific time. Both equations use the kinematic acceleration equations (5.9) which are obtained by differentiating the constraint equations (5.2) twice with respect to time. This means that the following differential equation is also being integrated with respect to time:

$$\ddot{\mathbf{\Phi}}(\mathbf{q}, t) \equiv \mathbf{\Phi_q}\,\ddot{\mathbf{q}} + \dot{\mathbf{\Phi}}_{\mathbf{q}}\,\dot{\mathbf{q}} + \dot{\mathbf{\Phi}}_t = \mathbf{0} \tag{5.19}$$

This system of differential equations has the following general solution:

$$\mathbf{\Phi}(\mathbf{q}, t) = \mathbf{a}_1 t + \mathbf{a}_2 \tag{5.20}$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ are constant vectors that depend on the initial conditions; that is, on the value of the constraint equations and on its first derivative with respect to time at t=0. If the position and initial velocity satisfy the constraint equations, both vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ are null. Theoretically, equation (5.20) guarantees that the constraint equations will be satisfied at any time. The fact of the matter is very different.

Equation (5.19) is *unstable*, since for any vector $\mathbf{a}_1$ different from zero the general solution given by equation (5.20) is not bounded and tends to increase indefinitely with time. Even though the initial conditions guarantee that $\mathbf{a}_1 = \mathbf{0}$, during the course of the numerical integration, the round-off errors that appear during the integration do not satisfy the constraint equations. The effects of these errors increase with time, in accordance with expression (5.20). Therefore, the

constant distances cease to be constant and the points of the same element progressively move closer to or further away from them. A similar situation happens with the other constraint equations.

This difficulty takes place during the integration of the differential equation (5.19). For example, let's take a look at a constraint equation of constant distance between two points i and j:

$$(\mathbf{r}_i - \mathbf{r}_j)^{\mathrm{T}} (\mathbf{r}_i - \mathbf{r}_j) - L_{ij}^2 = 0 \tag{5.21}$$

By differentiating this equation twice with respect to time, one obtains

$$(\mathbf{r}_i - \mathbf{r}_j)^{\mathrm{T}} (\ddot{\mathbf{r}}_i - \ddot{\mathbf{r}}_j) + (\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j)^{\mathrm{T}} (\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j) = 0 \tag{5.22}$$

When numerically integrating equation (5.22), one loses the information about the distance L_{ij} that should be maintained between both of the points. The constant L_{ij} disappears during the differentiating process. Thus, expression (5.22) does not have the information corresponding to the distance that must be maintained, and errors accumulate on the distance between points i and j. The same situation happens with all the other constraint equations, which at the time of differentiation lose the information carried by the constant terms.

The instability during the numeric integration of kinematic acceleration equations ensures that Algorithms 5-1 and 5-2 may not be used directly to obtain the solution of the dynamic simulation problem. Two different methods have been proposed for overcoming this difficulty and are stated below.

5.1.3.1 Integration of a Mixed System of Differential and Algebraic Equations.

The purpose of this method is to jointly solve the system of nonlinear algebraic equations (5.1) and differential equations (5.10) or (5.17). Thus, the constraint equations, and not only their second derivatives, are satisfied at any given time. Refer to Chapter 7 for a description on the general solution of differential algebraic equations or DAEs. There are numerical integration methods for mixed systems of differential equations that permit adding algebraic equations. However, they are not the most numerically efficient and are not free from stability problems in the simulation of mechanical systems (See Chapter 7 and Steigerwald (1990)).

5.1.3.2 Baumgarte Stabilization

The aim of the Baumgarte stabilization method (Baumgarte (1972)) is to replace the differential constraint equations (5.19) by the following system:

$$\ddot{\mathbf{\Phi}} + 2\alpha \dot{\mathbf{\Phi}} + \beta^2 \mathbf{\Phi} = \mathbf{0} \tag{5.23}$$

where α and β are appropriately chosen constants. The general solution to this differential equation is

$$\mathbf{\Phi} = \mathbf{a}_1 \, \mathrm{e}^{s_1 t} + \mathbf{a}_2 \, \mathrm{e}^{s_2 t} \tag{5.24}$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ are constant vectors that depend on the initial conditions, and s_1 and s_2 are the roots of the characteristic equation, defined by the expression:

$$s_1, s_2 = -\alpha \pm \sqrt{\alpha^2 - \beta^2} \tag{5.25}$$

If α and β are positive constants, the two roots s_1 and s_2 have a real negative part which guarantees the stability of the general solution (5.24) in contrast with that in (5.20). The initial position and velocity conditions of the multibody system should guarantee that the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ are zero. If the numerical rounding-off errors alter this condition, the real negative part of the exponential terms damps out the possible errors occurring during the integration process. The constants α and β are usually equal to one another with values between 1 and 20, and it appears that the behavior of the method does not significantly depend on these values. Chang and Nikravesh (1985), and Bae and Yang (1990) proposed different methods for optimizing this choice.

By using equation (5.23) instead of equation (5.9), the differential equations of motion (5.10) and (5.17) are respectively transformed into:

$$\begin{bmatrix} \mathbf{M} & \mathbf{\Phi}_\mathbf{q}^\mathrm{T} \\ \mathbf{\Phi}_\mathbf{q} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ \mathbf{g} \end{Bmatrix} \tag{5.26}$$

and

$$\begin{bmatrix} \mathbf{\Phi}_\mathbf{q} \\ \mathbf{R}^\mathrm{T}\mathbf{M} \end{bmatrix} \ddot{\mathbf{q}} = \begin{Bmatrix} \mathbf{g} \\ \mathbf{R}^\mathrm{T}\mathbf{Q} \end{Bmatrix} \tag{5.27}$$

where

$$\mathbf{g} = -\dot{\mathbf{\Phi}}_t - \dot{\mathbf{\Phi}}_\mathbf{q}\,\dot{\mathbf{q}} - 2\alpha\left(\mathbf{\Phi}_\mathbf{q}\,\dot{\mathbf{q}} + \mathbf{\Phi}_t\right) - \beta^2\,\mathbf{\Phi} \tag{5.28}$$

Nikravesh (1984) has studied comparatively the numerical integration of the equations of motion with dependent coordinates without stabilization, with Baumgarte stabilization, and integrating mixed systems of differential and algebraic equations. His conclusions indicate that the Baumgarte stabilization is twice as efficient as the integration of the mixed systems, even though not all the problems examined were satisfactorily solved with the said stabilization method. On the other hand, the direct integration of equations (5.10) or (5.17) produced unacceptable results. The Baumgarte stabilization is general, simple, and numerically efficient. Its computational cost is a small fraction of the total required. However, it does not solve all possible instabilities, such as near kinematic singular configurations (Haug (1989)). This aspect works in favor of other methods with dependent and independent coordinates that will be studied below.

5.1.4 Penalty Formulations

As shown in Section 5.1.1, the Lagrange multipliers technique allows for the solution of the dynamic problem at the expense of solving for an augmented set of

(n+m) unknowns: $\mathbf{q}$ plus $\boldsymbol{\lambda}$. In this section we will present an alternative penalty formulation proposed by Bayo et al. (1988) that eliminates the Lagrange multipliers from the equations of motion and leads to a set of n ordinary differential equations with $\ddot{\mathbf{q}}$ as the only unknowns. In essence, this method directly incorporates the constraint equations as a dynamical system, penalized by a large factor, into the equations of motions. The larger the penalty factor the better the constraints will be achieved at the cost of introducing some numerical ill-conditioning. We will show next how this penalty formulation can be applied to holonomic and non-holonomic constraint conditions and how to avoid the numerical problems that may arise in the use of penalty factors. Theoretical studies of its convergence and stability have been carried out by Kurdila and Narcowich (1992). This penalty method has also been successfully extended to real time dynamics within the context of fully Cartesian coordinates in Bayo et al. (1991). This will be explained in Chapter 8.

Holonomic Systems. A holonomic system is characterized by constraint equations of the form given in (5.2) which represent a set of nonlinear algebraic equations in the coordinates and the time variable. The penalty formulation is derived by adding three terms to the *Lagrangian*: These terms include a fictitious potential:

$$V^* = \sum_k \frac{1}{2}\,\alpha_k\,\omega_k^2\,\Phi_k^2 \equiv \frac{1}{2}\,\boldsymbol{\Phi}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\boldsymbol{\Omega}^2\,\boldsymbol{\Phi} \tag{5.29}$$

a set of Rayleigh's dissipative forces:

$$G_k = -\,2\,\alpha_k\,\omega_k\,\mu_k\,\frac{d\phi_k}{dt} \equiv -\,2\,\boldsymbol{\alpha}\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}} \tag{5.30}$$

and a fictitious kinetic energy term:

$$\times \qquad T^* = \sum_k \frac{1}{2}\,\alpha_k \left(\frac{d\Phi_k}{dt}\right)^2 \equiv \frac{1}{2}\,\dot{\boldsymbol{\Phi}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\dot{\boldsymbol{\Phi}} \tag{5.31}$$

The α_k are very large real values (penalty numbers), and ω_k and μ_k represent the natural frequency and the damping ratio of the penalty system (mass, dashpot, and spring) corresponding to the constraint Φ_k=*0*. Matrices $\boldsymbol{\alpha}$, $\boldsymbol{\Omega}$ and $\boldsymbol{\mu}$ are (m×m) diagonal matrices that contain the values of the penalty numbers, the natural frequencies, and the damping ratios of the penalty systems assigned to each constraint condition. If the same values are used for each constraint, these matrices become identity matrices multiplied by the respective penalty numbers. Note that in equations (5.29) through (5.31), we have used both index as well as matrix notation, hoping that this will lead to a better understanding of the physical significance of the different terms. In the following discussion, we will only use the matrix form in order to be consistent with the notation used so far in this book.

The differentiation of the new penalty terms that form the Lagrangian term L^*=T^*–V^* leads to

$$\frac{\partial L^*}{\partial \mathbf{q}} = \dot{\mathbf{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \dot{\mathbf{\Phi}} - \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \boldsymbol{\Omega}^2\, \mathbf{\Phi} \tag{5.32}$$

$$\frac{\partial L^*}{\partial \dot{\mathbf{q}}} = \dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \dot{\mathbf{\Phi}} \tag{5.33}$$

$$\frac{d}{dt}\left(\frac{\partial L^*}{\partial \dot{\mathbf{q}}}\right) = \ddot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \dot{\mathbf{\Phi}} + \dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \ddot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \dot{\mathbf{\Phi}} + \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \ddot{\mathbf{\Phi}} \tag{5.34}$$

where the easily verifiable relation $\ddot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{\mathrm{T}} = \dot{\mathbf{\Phi}}_{\mathbf{q}}^{\mathrm{T}}$ is used.

The work done by the fictitious Rayleigh forces is

$$\delta W_R = -\,2\,(\delta \mathbf{\Phi})^{\mathrm{T}}\, \boldsymbol{\alpha}\, \boldsymbol{\Omega}\, \boldsymbol{\mu}\, \dot{\mathbf{\Phi}} = -\,2\,\delta \mathbf{q}^{\mathrm{T}}\, \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \boldsymbol{\Omega}\, \boldsymbol{\mu}\, \dot{\mathbf{\Phi}} \tag{5.35}$$

and, therefore, the final expression obtained by the application of the Lagrange's equations is

$$\mathbf{M}\, \ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \left(\ddot{\mathbf{\Phi}} + 2\, \boldsymbol{\Omega}\, \boldsymbol{\mu}\, \dot{\mathbf{\Phi}} + \boldsymbol{\Omega}^2\, \mathbf{\Phi}\right) = \mathbf{Q} \tag{5.36}$$

where $\mathbf{M}$ and $\mathbf{Q}=\mathbf{Q}_{\mathrm{ex}}+L_{\mathbf{q}}-\dot{\mathbf{M}}\dot{\mathbf{q}}$ are the mass matrix and the force vector corresponding to the system without constraints.

Remark: The second term in the LHS of equation (5.36) represents the forces that are generated by the penalty system when the constraints $\mathbf{\Phi}$, $\dot{\mathbf{\Phi}}$, and $\ddot{\mathbf{\Phi}}$ are violated. The virtual power method leads to this result directly without the need of differentiation, as is the case with the Lagrange's equations. By merely comparing equations (5.36) with (5.4), we may see that ($\boldsymbol{\alpha}\mathbf{\Phi}+2\boldsymbol{\Omega}\boldsymbol{\mu}\dot{\mathbf{\Phi}}+\boldsymbol{\Omega}^2\ddot{\mathbf{\Phi}}$) is an approximation to the true Lagrange multipliers $\boldsymbol{\lambda}$. The pre-multiplication by $\mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}$ projects these forces unto the space of the dependent coordinates.

Substituting for $\ddot{\mathbf{\Phi}}$ with values from equation (5.9) the following final result is obtained:

$$(\mathbf{M} + \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \mathbf{\Phi}_{\mathbf{q}})\, \ddot{\mathbf{q}} = \mathbf{Q} - \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\alpha}\, (\dot{\mathbf{\Phi}}_{\mathbf{q}}\, \dot{\mathbf{q}} + \dot{\mathbf{\Phi}}_t + 2\, \boldsymbol{\Omega}\, \boldsymbol{\mu}\, \dot{\mathbf{\Phi}} + \boldsymbol{\Omega}^2\, \mathbf{\Phi}) \tag{5.37}$$

It has been demonstrated (Oden (1983)) that the solution of the modified problem coincides with that of the original problem provided that $\alpha_k \to \infty$. Numerically, this condition is achieved by merely using large penalty factors. These in turn may produce numerical ill conditioning which may be avoided by the improved technique described below. For double precision arithmetic, a factor of 10^7 times the largest term of the mass matrix gives excellent results. The coefficients ω and μ may have a stability effect similar to that produced by the α and β coefficients of the Baumgarte constraint stabilization method explained above. However, the penalty formulation of equation (5.37) does not fail near kinematic singular configurations or with redundant constraints, as the Baumgarte stabilization does.

Non-holonomic Systems. The penalty formulation also allows one to introduce, with no difficulty, non-holonomic constraints. The general form of a non-holonomic constraint is

$$\Phi_k(q_j, \dot{q}_j, t) = 0 \tag{5.38}$$

Non-holonomic constraints conditions for multibody systems typically take the form:

$$\boldsymbol{\Phi} = \mathbf{A}(\mathbf{q}, t)\, \dot{\mathbf{q}} + \mathbf{B}(\mathbf{q}, t) \tag{5.39}$$

The classical Lagrange multiplier approach leads to the following equations of motion (Goldstein (1980)):

$$\mathbf{M}\, \ddot{\mathbf{q}} + \mathbf{A}^{\mathrm{T}} \boldsymbol{\lambda} = \mathbf{Q} \tag{5.40}$$

Considering the penalty formulation, we introduce as for the holonomic systems a set of fictitious Rayleigh's dissipative forces that are proportional to the velocities:

$$G_k^* = -\, \alpha_k\, \mu_k\, \phi_k \equiv -\, \boldsymbol{\alpha}\, \boldsymbol{\mu}\, \boldsymbol{\Phi} \tag{5.41}$$

and of inertia forces:

$$I_k^* = -\, \alpha_k\, \dot{\phi}_k \equiv -\, \boldsymbol{\alpha}\, \dot{\boldsymbol{\Phi}} \tag{5.42}$$

The projection of the forces acting on the constraints over the space of dependent coordinates is given by

$$\mathbf{A}^{\mathrm{T}}\, \boldsymbol{\alpha}\, (\dot{\boldsymbol{\Phi}} + \boldsymbol{\mu}\, \boldsymbol{\Phi}) \tag{5.43}$$

and the application of the virtual power method directly leads to

$$\mathbf{q}^{*\mathrm{T}} \left(\mathbf{M}\, \ddot{\mathbf{q}} - \mathbf{Q} + \mathbf{A}^{\mathrm{T}}\, \boldsymbol{\alpha}\, (\dot{\boldsymbol{\Phi}} + \boldsymbol{\mu}\, \boldsymbol{\Phi}) \right) = 0 \tag{5.44}$$

Since the penalty formulation makes the problem become unconstrained, the virtual velocities may be arbitrarily selected. The expression between brackets vanishes, consequently,

$$\mathbf{M}\, \ddot{\mathbf{q}} + \mathbf{A}^{\mathrm{T}}\, \boldsymbol{\alpha}\, (\dot{\boldsymbol{\Phi}} + \boldsymbol{\mu}\, \boldsymbol{\Phi}) = \mathbf{Q} \tag{5.45}$$

Knowing that

$$\dot{\boldsymbol{\Phi}}(\mathbf{q}, \dot{\mathbf{q}}, t) = \boldsymbol{\Phi}_{\mathbf{q}}\, \dot{\mathbf{q}} + \boldsymbol{\Phi}_{\dot{\mathbf{q}}}\, \ddot{\mathbf{q}} + \boldsymbol{\Phi}_t \tag{5.46}$$

equation (5.45) becomes

$$(\mathbf{M} + \mathbf{A}^{\mathrm{T}}\, \boldsymbol{\alpha}\, \mathbf{A})\, \ddot{\mathbf{q}} = \mathbf{Q} - \mathbf{A}^{\mathrm{T}}\, \boldsymbol{\alpha}\, (\dot{\mathbf{A}}\, \dot{\mathbf{q}} + \dot{\mathbf{B}} + \boldsymbol{\mu}\, \boldsymbol{\Phi}) \tag{5.47}$$

As in the holonomic case, the diagonal matrix $\boldsymbol{\alpha}$ can be substituted by a constant that multiplies the identity matrix, if the same penalty number is used for all of the constraints.

Augmented Lagrangian Formulation. Equations (5.37) and (5.47) form the modified Lagrange's equations that are obtained by virtue of a penalty formulation. Penalty methods bring forth the problem of choosing the right penalty number. While large penalty values will ensure convergence to the constraint within a tight tolerance, those values may also lead to numerical conditioning problems and develop round-off errors. It is therefore important that the analyst be supplied with a method that converges, regardless of the size of the penalty values, to the right solution within specified tolerances in the constraints. This method will have all the possible advantages of a reduced number of equations and have no need for very large penalty values to assure convergence.

To this end, we can extend the augmented Lagrangian method commonly used in optimization analysis (Vanderplaats (1984)) to improve the numerical conditioning of the proposed penalty equations. Let us consider again the classical Lagrange multipliers method as stated by equation (5.8). This method, along with the constraints (5.1), forms a system of DAEs whose solution will yield the values of the n generalized coordinates q_j as well as the m Lagrange multipliers λ_k. Instead of following this approach, we can modify equation (5.8) by adding the corresponding penalty terms, whose values will be zero if the constraints are satisfied. Therefore

$$\mathbf{M}\,\ddot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\ddot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}} + \boldsymbol{\Omega}^2\,\boldsymbol{\Phi}\right) + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\lambda}^{*} = \mathbf{Q} \tag{5.48}$$

This new equation can be viewed as a penalty method to which the Lagrange multipliers are added. In the limit, the constraint conditions are satisfied; thus $\boldsymbol{\lambda}=\boldsymbol{\lambda}^{*}$ and equations (5.8) and (5.48) become totally equivalent except for round-off errors induced by the penalty parameters and finite machine precision. In (5.48) the Lagrange multipliers $\boldsymbol{\lambda}^{*}$ play the role of correcting terms.

By merely comparing equations (5.8) and (5.48), it can be inferred that

$$\boldsymbol{\lambda} \cong \boldsymbol{\lambda}^{*} + \boldsymbol{\alpha}\left(\ddot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}} + \boldsymbol{\Omega}^2\,\boldsymbol{\Phi}\right) \tag{5.49}$$

We are seeking the solution of (5.48) without having to use the algebraic constraint equations (5.1). This requires that the correct values of $\boldsymbol{\lambda}^{*}$ be known, so that they can be inserted in (5.49). Since those values are not known in advance, there is a need to set up an iterative process that calculates the unknown multipliers $\boldsymbol{\lambda}^{*}$. The iteration is easily established by taking advantage of equation (5.49):

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_{i} + \boldsymbol{\alpha}\left(\ddot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}} + \boldsymbol{\Omega}^2\,\boldsymbol{\Phi}\right)_{i+1} \qquad i = 0,\ 1,\ 2,\ \ldots. \tag{5.50}$$

with $\boldsymbol{\lambda}_0^{*}=0$ for the first iteration. Equation (5.50) physically represents the introduction at iteration $i+1$ of forces that tend to compensate the fact that the constraints are not exactly zero. It becomes obvious now that the penalty number does not need to be very large, since the resulting error in the constraint equations will be eliminated by the Lagrange's terms during the iteration procedure. The

generic penalty method corresponds to the augmented Lagrangian method in which the iteration process is carried out only once.

The matrix formulation of (5.48), including the iterative process defined in (5.50), is given by the following expression:

$$\begin{gathered}\left(\mathbf{M}+\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\boldsymbol{\Phi}_{\mathbf{q}}\right)\ddot{\mathbf{q}}_{\mathrm{i}+1}= \\ =\mathbf{M}\,\ddot{\mathbf{q}}_{\mathrm{i}}-\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\dot{\boldsymbol{\Phi}}_{\mathbf{q}}\,\dot{\mathbf{q}}+\dot{\boldsymbol{\Phi}}_{t}+2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}}+\boldsymbol{\Omega}^{2}\,\boldsymbol{\Phi}\right) \\ i=0,\,1,\,2,\,\ldots\end{gathered} \tag{5.51}$$

with $\mathbf{M}\,\ddot{\mathbf{q}}_0=\mathbf{Q}$ for the initial iteration. The subscript i represents the iteration number.

This improved formulation for the non-holonomic case leads to an iterative procedure as given by the following equation:

$$\begin{gathered}\left(\mathbf{M}+\mathbf{A}^{\mathrm{T}}\boldsymbol{\alpha}\,\mathbf{A}\right)\ddot{\mathbf{q}}_{\mathrm{i}+1}=\mathbf{M}\,\ddot{\mathbf{q}}_{\mathrm{i}}-\mathbf{A}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\dot{\mathbf{A}}\,\dot{\mathbf{q}}+\dot{\mathbf{B}}+\boldsymbol{\mu}\,\boldsymbol{\Phi}\right) \\ \mathrm{i}=0,\,1,\,2,\,\ldots\end{gathered} \tag{5.52}$$

with again $\mathbf{M}\,\ddot{\mathbf{q}}_0=\mathbf{Q}$ for the initial iteration.

At first, this procedure might seem to be at a disadvantage since an iteration process and thus extra computation are required. However, the extra numerical effort is practically insignificant, since an iterative procedure is usually necessary to solve a system of nonlinear differential equations. A major advantage obtained in return for this additional computation is that the analyst does not have to be concerned with the value of the penalty number that simultaneously assures convergence and avoids round-off errors. The numerical integration of the equations of motion using the penalty formulation may proceed as follows:

Algorithm 5-3

1. Start at a time t, when the position $\mathbf{q}$ and velocity $\dot{\mathbf{q}}$ are known.
2. Use equation (5.37) in holonomic systems, or (5.47) in non-holonomic systems, to solve for the accelerations $\ddot{\mathbf{q}}$ at time t. If the augmented Lagrangian is desired, then use equations (5.52) and (5.53) for holonomic and non-holonomic systems, respectively.
3. Obtain the vectors $\mathbf{q}$ and $\dot{\mathbf{q}}$ at time $(t+\Delta t)$ by numerical integration:

$$\dot{\mathbf{y}}_t^{\mathrm{T}}\equiv\{\ddot{\mathbf{q}}^{\mathrm{T}},\dot{\mathbf{q}}^{\mathrm{T}}\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}_{t+\Delta t}^{\mathrm{T}}\equiv\{\dot{\mathbf{q}}^{\mathrm{T}},\mathbf{q}^{\mathrm{T}}\}_{t+\Delta t} \tag{5.53}$$

4. Upon convergence of the n.i.s., update the time variable and go to step 2.

This numerical integration algorithm has the advantage of solving a set of n equations as compared to $(n+m)$ needed by the Lagrange multiplier method. Constraint stabilization is implicitly considered within the algorithm and is implemented more simply than the methods that use independent coordinates which are shown in the sequel. An efficient numerical implementation of this penalty method, which is more suitable for real time applications, has been proposed by Bayo et al. (1991) and will be explained in detail in Chapter 8.

Table 5.1. Maximum constraint errors for different penalty values.

	Number of iterations					
	0		1		2	
Penalty	time	error	time	error	time	error
10^1	10.5	$0.7\ 10^{-1}$	12.8	$0.7\ 10^{-2}$	14.1	$0.9\ 10^{-3}$
10^3	11.7	$0.7\ 10^{-3}$	13.3	$0.6\ 10^{-5}$	14.6	$0.6\ 10^{-5}$
10^5	11.7	$0.1\ 10^{-4}$	13.3	$0.6\ 10^{-5}$	14.6	$0.6\ 10^{-5}$
10^7	11.7	$0.6\ 10^{-5}$	13.3	$0.6\ 10^{-5}$	14.6	$0.6\ 10^{-5}$
10^9	11.7	$0.6\ 10^{-5}$	13.3	$0.6\ 10^{-5}$	14.6	$0.6\ 10^{-5}$
10^{11}	17.3	$0.1\ 10^{-4}$	19.1	$0.9\ 10^{-5}$	21.9	$0.2\ 10^{--4}$

Example 5.1

Given the results of a numerical simulation of the motion of a double pendulum moving in a vertical plane under gravitational forces. The pendulum has two elements of unit mass and length. Four natural coordinates with two constraints conditions are used to model the system. We use the penalty-augmented Lagrangian method with coefficients Ω and μ equal to 10 and 1, respectively. These provide critical damping in the stabilization process. Table 5.1 contains a comparative study of the resulting maximum constraint errors and CPU times in seconds obtained, using different penalty numbers and 0, 1 and 2 iterations. In all the cases, the integration is performed using the subroutine DGEAR (Gear (1971)) with an error tolerance equal to 10^{-4}. It may be seen how the use of only one iteration considerably widens the range of acceptable penalty values at practically no additional computational cost.

5.2 Formulations in Independent Coordinates

Some of the methods used to formulate and integrate the motion differential equations in independent coordinates will be presented below. One advantage of this type of coordinates is an important reduction in the number of equations to be integrated. Most important is the disappearance of the instability problem in the integration of the constraint equations using ODE solvers. However, this has a price in terms of computational effort since the position and velocity problems need to be solved after the function evaluations. Some of the numerical integration algorithms studied in Chapter 7, and in particular the more stable implicit algorithms are difficult to implement. In addition, the formulation and implementation of these methods become more involved than those which use dependent

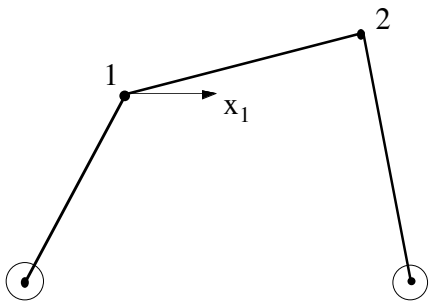

Figure 5.1. Independent velocity in a four-bar mechanism.

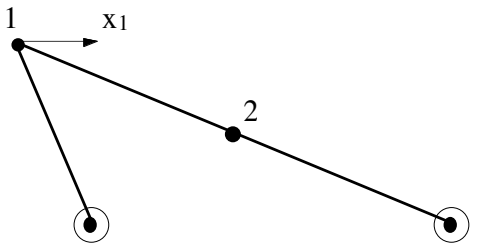

Figure 5.2. Need to change the independent velocity in a four-bar mechanism.

coordinates. One important point is the choice of the right set of independent coordinates. This point will be studied in greater detail next.

5.2.1 Determination of Independent Coordinates

This is a point of transcendental importance. The independent velocities normally are given by the projection of the dependent velocities $\ddot{\mathbf{q}}$ on certain vectors defined by the rows of a constant matrix **B** (See Section 3.5) as:

$$\dot{\mathbf{z}} = \mathbf{B}\,\dot{\mathbf{q}} \tag{5.54}$$

The need to suitably select the independent coordinates can be illustrated from the mechanical point of view with some very simple mechanisms. For example, in the four-bar mechanism of Figure 5.1, the velocity $\dot{x}_1$ perfectly defines all the mechanism's velocities. However, in the quadrilateral of Figure 5.2, this coordinate is not adequate, since in no way does it determine the velocity of point 2. In fact, at the position of Figure 5.2, $\dot{x}_1$ will always be zero.

Figures 5.3 and 5.4 show a mechanism with five bars at two positions. At one of these the selected coordinates are adequate, but at the other they are not. When bars 2 and 3 are parallel, bars 3 and 4 have the possibility of being jointly moved as a rigid body. This motion is not determined by angles ψ_1 and ψ_2.

The examples mentioned should be sufficient for understanding: first, no system of independent coordinates is adequate for the entire motion of the system;

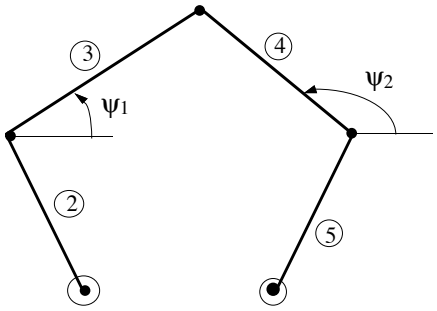

Figure 5.3. Adequate set of independent coordinates in a five-bar mechanism.

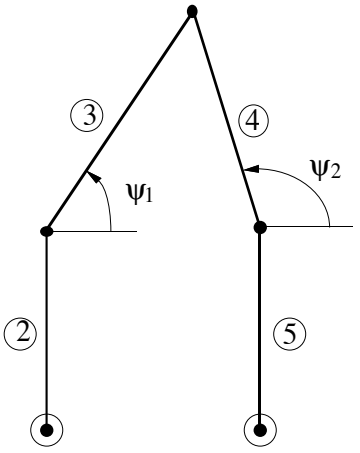

Figure 5.4. Inadequate independent coordinates in a five-bar mechanism.

and secondly, that a particular set of independent coordinates may be the most adequate at a certain position of the mechanism but not at another. Therefore, one must establish a double actuation procedure. On one hand a method must be developed that permits checking when a set of independent coordinates is becoming inadequate. On the other hand, it is necessary to establish a method for finding the most adequate new set of independent coordinates. Fortunately, there are mathematical properties of the Jacobian matrix $\mathbf{\Phi_q}$ that permit the solution of the two problems satisfactorily. These properties will be seen later in connection with the specific methods of formulating and solving the dynamic problem with independent coordinates.

One last important point is that normally the numerical integration subroutines of ordinary differential equations are based on multistep methods (Shampine and Gordon (1975); Gear (1971)). These methods are very efficient, but they have certain limitations. They require special techniques for starting the integration process, and they are long and drawn out. Since it is necessary to change each time the independent coordinates, the numerical integration must be restarted again. One must carry out the minimum possible number of coordinate changes. On the other hand, when some determined coordinates start to be inadequate, the integration process becomes much slower. It is necessary to arrive at a compromise solution, therefore, by making the minimum number of coordinate changes that guarantee quick and accurate numerical integration. Sometimes, the speed of the numerical integration can be utilized as the criteria for the change of independent coordinates, when subroutines with automatic step size control are used.

The most important numerical integration methods for the differential equations of motion in independent coordinates are described next.

5.2.2 *Extraction Methods (Coordinate Partitioning)*

This method, proposed by Wehage and Haug (1982), consists of finding the dependent accelerations $\ddot{\mathbf{q}}$ by means of equations (5.10) or (5.17)Only some of the vector's elements are integrated, specifically, those corresponding to the independent coordinates. Wehage and Haug have called this technique the *coordinate partitioning method.*

In order to choose an independent set of coordinates from vector $\mathbf{q}$, it should be remembered how the numerical integration subroutine behaves in this case. Consider the vector $\mathbf{q}$ and a partition of dependent and independent coordinates as follows:

$$\mathbf{q}^{\mathrm{T}} = \{\mathbf{q}_{\mathrm{d}}^{\mathrm{T}}, \mathbf{q}_{\mathrm{i}}^{\mathrm{T}}\} \tag{5.55}$$

where there are m dependent coordinates and $(f=n-m)$ independent coordinates.

The numerical integration subroutine is applied to only the independent coordinates as follows:

$$\dot{\mathbf{y}}_t^{\mathrm{T}} \equiv \{\ddot{\mathbf{q}}_{\mathrm{i}}^{\mathrm{T}}, \dot{\mathbf{q}}_{\mathrm{i}}^{\mathrm{T}}\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}_{t+\Delta t}^{\mathrm{T}} \equiv \{\dot{\mathbf{q}}_{\mathrm{i}}^{\mathrm{T}}, \mathbf{q}_{\mathrm{i}}^{\mathrm{T}}\}_{t+\Delta t} \tag{5.56}$$

Since only the independent coordinates and velocities at time $(t+\Delta t)$ are known, it is necessary to calculate the remaining coordinates and velocities. This is done by solving the position problem to calculate $\mathbf{q}_{\mathrm{d}}$ in terms of $\mathbf{q}_{\mathrm{i}}$ after each function evaluation, and doing the same for the velocities. The latter requires the solution of the following set of linear equations which are the partitioned constraint equations for velocities:

$$\left[\, \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{d}} \;\; \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{i}} \,\right] \begin{Bmatrix} \dot{\mathbf{q}}_{\mathrm{d}} \\ \dot{\mathbf{q}}_{\mathrm{i}} \end{Bmatrix} = \mathbf{b} \tag{5.57}$$

where the partition carried out on the vector $\ddot{\mathbf{q}}$ leads to a similar partition on the Jacobian matrix $\boldsymbol{\Phi}_{\mathbf{q}}$. In order to calculate the dependent variables, the inverse of matrix $\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{d}}$ must exist. During the Gauss triangularization process of this matrix with column pivoting to maintain the previously determined partition, all the pivots must be sufficiently different from zero. The fact that one or more of the pivots of $\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{d}}$ tend to zero, means that the current set of independent coordinates are becoming inadequate. More specifically, the dependent velocity corresponding to the column in which the pivot tends towards zero appears, must now be taken as a new independent coordinate.

When a system of independent coordinates becomes inappropriate in actual practice, rather than substituting one coordinate for another, it is recommended that one choose a new complete system of independent coordinates. This is done by carrying out the factorization of $\boldsymbol{\Phi}_{\mathbf{q}}$ with total pivoting. The $(n-m)$ columns in which the m pivots have not appeared will determine the coordinates that should be chosen as independent ones.

Although Gauss total pivoting is neither the only nor the most reliable technique that may be used for determining a decomposition of vector $\mathbf{q}$ into depen-

dent and independent coordinates, it is undoubtedly the most economical. Mani et al. (1985) proposed the Singular Value decomposition (SV) and Kim and Vanderploeg (1986b) the QR decomposition. Both are, without a doubt, more reliable than the Gauss total pivoting method, but require considerably more calculation effort. These techniques can be used perhaps to choose a new set of independent coordinates at specific positions, because this process will only need to be carried out very few times in all the simulation. However, the QR or SV decompositions are completely unsuitable to detect the need for change in the set of independent coordinates. Since the detection must be carried out at each step of the numerical integration, these decompositions would unacceptably delay the integration process.

This numerical integration process requires solving the position problem and performing the velocity analysis at each iteration. The latter does not constitute an important difficulty. However, the position problem does, because it requires an iterative solution that consumes an important amount of computational time. For this reason, Paul (1975) suggested the integration of the following extended set of differential equations:

$$\dot{\mathbf{y}}_t^T \equiv \{\ddot{\mathbf{q}}_i^T, \dot{\mathbf{q}}^T\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}_{t+\Delta t}^T \equiv \{\dot{\mathbf{q}}_i^T, \mathbf{q}^T\}_{t+\Delta t} \tag{5.58}$$

By integrating all the velocities, and not only the independent ones, the new position of the multibody system is directly obtained as a result of the numerical integration. With numerical integration, the constraint equation stabilization problem is not so critical. The equation that is integrated is that of the velocities instead of accelerations. The general solution of the first derivative of the constraint equations is simply a constant vector. Therefore, round-off errors do not tend to increase with time, although they accumulate and slow down the integration.

Numerical integration of the extended differential equations system (5.58) is frequently used. Generally speaking it is more efficient than that of system (5.56) which, as mentioned above, entails a repeated solution of the position and velocity problem at each step. For long simulations, if the use of (5.58) leads to an accumulation of constraint errors which may even lead to numerical stiffness in the solution of equations, then the position problem is solved and the integration proceeds. Park and Haug (1986) proposed a hybrid method that combines the coordinate partitioning and Baumgarte stabilization of the constraints. Thus, when the errors in the constraint exceed a specified tolerance, the accelerations are calculated with equation (5.26) instead of (5.10). When the Baumgarte stabilization fails in the neighborhood of a kinematically singular configuration, the proposed method reverts to pure coordinate partitioning with the solution of the position and velocity problems.

To summarize, the extraction method calculates all the dependent accelerations with the same formulae used in the methods based on dependent coordinates. It then integrates only one subset of the accelerations chosen by means of Gauss triangularization with total pivoting or by the QR or SV decomposition. To determine when an independent coordinates system starts becoming inadequate, the

best and most economical method is to check the pivots during the triangularization with column pivoting. This should be done at each step of the numerical integration.

5.2.3 *Methods Based on the Projection Matrix* **R**

One should recall the velocity equation, obtained after differentiating the constraint equations with respect to time:

$$\mathbf{\Phi_q}\,\dot{\mathbf{q}} = -\mathbf{\Phi}_t \equiv \mathbf{b} \tag{5.59}$$

Likewise one should consider again equation (5.54), in which the independent velocities $\dot{\mathbf{z}}$ are defined as the projection of vector $\dot{\mathbf{q}}$ on the rows of a constant matrix $\mathbf{B}$ of size $((n-m)\times n)$. The rows of matrix $\mathbf{B}$ satisfy the condition of being linearly independent from one another and with respect to the rows of matrix $\mathbf{\Phi_q}$. By jointly writing the expressions (5.59) and (5.54) one obtains

$$\begin{bmatrix} \mathbf{\Phi_q} \\ \mathbf{B} \end{bmatrix} \dot{\mathbf{q}} = \begin{Bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{Bmatrix} \tag{5.60}$$

The matrix on the LHS of this expression is invertible and consequently,

$$\dot{\mathbf{q}} = \begin{bmatrix} \mathbf{\Phi_q} \\ \mathbf{B} \end{bmatrix}^{-1} \begin{Bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{Bmatrix} \equiv \begin{bmatrix} \mathbf{S} & \mathbf{R} \end{bmatrix} \begin{Bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{Bmatrix} = \mathbf{S}\,\mathbf{b} + \mathbf{R}\,\dot{\mathbf{z}} \tag{5.61}$$

where $\mathbf{S}$ and $\mathbf{R}$ are $(n\times m)$ and $(n\times(n-m))$ matrices, respectively. It is easy to demonstrate that term $\mathbf{R}\dot{\mathbf{z}}$ represents the general solution of the homogeneous velocity equation, and that the term $\mathbf{Sb}$ represents a particular solution of the complete equation. This is a particular solution of the velocity equation for the case in which there are rheonomous constraints.

By differentiating equations (5.59) and (5.54) with respect to time, one obtains

$$\mathbf{\Phi_q}\,\ddot{\mathbf{q}} = -\dot{\mathbf{\Phi}}_t - \dot{\mathbf{\Phi}}_{\mathbf{q}}\,\dot{\mathbf{q}} \equiv \mathbf{c} \tag{5.62}$$

$$\mathbf{B}\,\ddot{\mathbf{q}} = \ddot{\mathbf{z}} \tag{5.63}$$

By jointly expressing these equations:

$$\begin{bmatrix} \mathbf{\Phi_q} \\ \mathbf{B} \end{bmatrix} \ddot{\mathbf{q}} = \begin{Bmatrix} \mathbf{c} \\ \ddot{\mathbf{z}} \end{Bmatrix} \tag{5.64}$$

Solving for $\ddot{\mathbf{q}}$ and introducing the matrices $\mathbf{S}$ and $\mathbf{R}$ defined in (5.61), we obtain

$$\ddot{\mathbf{q}} = \begin{bmatrix} \mathbf{\Phi_q} \\ \mathbf{B} \end{bmatrix}^{-1} \begin{Bmatrix} \mathbf{c} \\ \ddot{\mathbf{z}} \end{Bmatrix} = \mathbf{S}\,\mathbf{c} + \mathbf{R}\,\ddot{\mathbf{z}} \tag{5.65}$$

It shall be remembered that matrix $\mathbf{R}$ must be explicitly calculated with $n-m$ forward and backward substitutions, starting from the leading matrix of (5.60)

factored by means of Gauss elimination. However, it is important to note that the matrix **S** never needs to be explicitly calculated. The terms (**Sb**) and (**Sc**) are, respectively, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, when $\dot{\mathbf{z}} = \mathbf{0}$ and $\ddot{\mathbf{z}} = \mathbf{0}$. These terms products can be directly calculated from expressions (5.60) and (5.64).

Eq. (5.16) represents the equations of motion with dependent coordinates:

$$\mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\ddot{\mathbf{q}} = \mathbf{R}^{\mathrm{T}}\,\mathbf{Q} \tag{5.66}$$

By introducing in this equation the equation (5.65) for the dependent accelerations in terms of the independent accelerations, we obtain

$$\mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{R}\,\ddot{\mathbf{z}} = \mathbf{R}^{\mathrm{T}}\,\mathbf{Q} - \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{S}\,\mathbf{c} \tag{5.67}$$

which constitutes the equations of motion in terms of independent coordinates. The derivation process that starts with equation (5.59) and ends in (5.67) leads to this general form of the equations of motion in independent coordinates, which was first introduced in this context by Serna et al. (1982). Equation (5.67) represents a general matrix transformation from the vector spaces of dependent accelerations and forces to the vector space of independent accelerations and forces.

This formulation is valid for both scleronomous and rheonomous constraint equations. In addition, this layout is valid, irrespective of the matrix **B** chosen in equation (5.54), provided that the conditions of being constant and of having its rows linearly independent from one another and with respect to the rows of $\mathbf{\Phi_q}$ are satisfied. The matrix **B** can be chosen in two different ways as explained next.

Boolean matrix. The matrix **B** is formed by a set of ones and zeros that extracts (*n*–*m*) components of **q** as independent coordinates **z**. By partitioning equation (5.60) in a similar way to that performed in equation (5.57), one obtains

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{5.68}$$

and

$$\begin{bmatrix} \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{d}} & \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{i}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}}_{\mathrm{d}} \\ \dot{\mathbf{q}}_{\mathrm{i}} \end{Bmatrix} = \begin{Bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{Bmatrix} \tag{5.69}$$

One should keep in mind that the matrix $\mathbf{\Phi}_{\mathbf{q}}^{\mathrm{d}}$ should be invertible in order to express the dependent velocities in terms of the independent ones. All the pivots must be sufficiently different from zero. In this way it is assured that the chosen rows of **B** are independent from those of $\mathbf{\Phi_q}$. If during the motion of a multibody system one of the pivots of $\mathbf{\Phi}_{\mathbf{q}}^{\mathrm{d}}$ becomes much smaller than the others, this independence is gradually lost. It is then necessary to choose new independent coordinates by means of a total pivoting process.

This is the simplest method of all those formulated in independent coordinates. In actual practice this method is almost always the most efficient one.

SV and QR Decompositions. As explained in Chapter 3, these methods take rows of matrix **B** as an orthogonal basis of the nullspace of $\mathbf{\Phi_q}$, calculated at a

previous position of the multibody system by means of the singular value (SV) or QR decompositions. Both lead to similar results. The SVD is more stable in poor numerically conditioned problems. However, the QR decomposition is numerically more efficient, since it is a direct process that does not need iterations.

The matrix $\mathbf{R}$ is also calculated starting from equation (5.60). As before, the matrix $\mathbf{B}$ is given by an orthonormal basis of the nullspace at a previous position. Since the row subspace of matrix $\mathbf{\Phi_q}$ is orthogonal to this nullspace, it is expected that the rows of $\mathbf{\Phi_q}$ are independent from the rows of $\mathbf{B}$ for a wide range of the motion. Since matrix $\mathbf{B}$ is kept constant, one may arrive at a position of the multibody system in which this independence is lost or deteriorated. Therefore, it will be necessary to change the independent coordinates by carrying out a new SV or QR decomposition. The most efficient method to find whether the rows of $\mathbf{B}$ are becoming linearly dependent is through the monitoring of the pivots during the Gauss triangularization process of equation (5.60).

Even though this method may occasionally require fewer changes of independent coordinates than the Boolean matrix method, the latter is simpler and numerically more efficient. It is not necessary to carry out additional operations to complete on $\mathbf{B}$ the Gauss triangularization of matrix $\mathbf{\Phi_q}$.

Numerical integration algorithm with projection matrices R. Of all the methods based on the projection matrix $\mathbf{R}$ shown in Chapter 3, the one based on the Boolean matrix $\mathbf{B}$ with rheonomous constraints will be chosen in order to present the corresponding numerical integration algorithm. The possibility exists of solving the position problem at each step, or integrating an enlarged system of differential equations. This second option is presented first.

Algorithm 5-4

1. Start at a time when the position $\mathbf{q}$ and velocities $\dot{\mathbf{z}}$ are known.
2. Calculate a new matrix $\mathbf{\Phi_q}$. Triangularize this matrix with column pivoting, and verify that all the pivots are sufficiently different from zero, so that the independent coordinates continue to be valid. Otherwise, carry out a new triangularization with total pivoting and choose a new set of independent coordinates. In addition, restart the numerical integration process if using a multistep method.
3. Form matrix $\mathbf{R}$ from equation (5.60). Note that the triangularization of $\mathbf{\Phi_q}$ has been carried out already in step 2.
4. Calculate the new dependent velocities $\dot{\mathbf{q}}$ by means of equation (5.61).
5. Form the matrix products $(\mathbf{R}^T\mathbf{M})$, $(\mathbf{R}^T\mathbf{M}\mathbf{R})$, and $(\mathbf{R}^T\mathbf{Q})$.
6. Calculate the terms $(\mathbf{Sb})$ and $(\mathbf{Sc})$ by making $\dot{\mathbf{z}} = 0$ and $\ddot{\mathbf{z}} = 0$ in equations (5.61) and (5.65), respectively.
7. Obtain the independent acceleration vector $\ddot{\mathbf{z}}$ from equation (5.67).
8. Obtain the vectors $\mathbf{q}$ and $\dot{\mathbf{z}}$ at time $(t+\Delta t)$ by numerical integration:

$$\dot{\mathbf{y}}_t^T \equiv \{\ddot{\mathbf{z}}^T, \dot{\mathbf{q}}^T\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}_{t+\Delta t}^T \equiv \{\dot{\mathbf{z}}^T, \mathbf{q}^T\}_{t+\Delta t} \tag{5.70}$$

9. Upon convergence of the n.i.s. update the time variable and go to step 2.

This algorithm constitutes an efficient and general purpose method of solving the forward dynamics. However, it is more difficult to implement than those based on dependent coordinates. By means of small modifications, this algorithm can be easily adapted to the other theoretically described projection methods. Since the integration is carried out using $\{\ddot{\mathbf{z}}^T, \dot{\mathbf{q}}^T\}$, errors in the constraint conditions may accumulate with the effect of slowing down the integration process. For long simulations and in order to eliminate this problem, it is necessary to solve the position problem either after a specified number of time steps or after checking at step 8 the fulfillment of the position constraint equations. In what follows we also give an algorithm based on $\{\ddot{\mathbf{z}}^T, \dot{\mathbf{z}}^T\}$ which requires the solution of the position and velocity problems in each iteration:

Algorithm 5-5

1. Start at a time in which the independent coordinates $\mathbf{z}$ and $\dot{\mathbf{z}}$ are known.
2. Solve the position and velocity problems to obtain $\mathbf{q}$ and $\dot{\mathbf{q}}$. Simultaneously, do the column pivoting on the matrix $\mathbf{\Phi_q}$ to check the validity of the current set of independent coordinates.
3. Form $\mathbf{R}$ from equation (5.60). Note that $\mathbf{\Phi_q}$ has already been triangularized.
4. Form the matrix products $(\mathbf{R}^T\mathbf{M})$, $(\mathbf{R}^T\mathbf{M}\mathbf{R})$, and $(\mathbf{R}^T\mathbf{M})$.
5. Calculate the terms $(\mathbf{Sb})$ and $(\mathbf{Sc})$ by making $\dot{\mathbf{z}} = 0$ and $\ddot{\mathbf{z}} = 0$ in equations (5.61) and (5.65), respectively.
6. Obtain the independent acceleration $\ddot{\mathbf{z}}$ from equation (5.67).
7. Obtain the vectors $\mathbf{z}$ and $\dot{\mathbf{z}}$ at time $(t+\Delta t)$ by the numerical integration:

$$\mathbf{y}'^T_t \equiv \{\ddot{\mathbf{z}}^T, \dot{\mathbf{z}}^T\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{y}^T_{t+\Delta t} \equiv \{\dot{\mathbf{z}}^T, \mathbf{z}^T\}_{t+\Delta t}$$

9. Upon convergence of the n.i.s., update the time variable and go to step 2.

5.2.4 *Comparative Remarks*

The penalty formulation characterized by equation (5.37) has the advantage over the formulations in independent coordinates, in that the appearance or disappearance of constraints can be accommodated automatically without changing the coordinates. This in turn avoids the restarting procedure of the numerical integrator. The penalty formulation is also more suitable when the multibody system goes through a singular or bifurcation position. In these cases the Jacobian matrix changes its rank, and the use of independent coordinates requires a sudden change of coordinates. Unless special provisions are made, the formulation in independent coordinates and even the Lagrange's equations in dependent coordinates tends to either crash the simulation or introduce sudden large errors. However, with the penalty formulation, the term $(\mathbf{M} + \mathbf{\Phi}_\mathbf{q}^T \boldsymbol{\alpha} \mathbf{\Phi}_\mathbf{q})$ of equation (5.37) is free of singularities and the integration becomes be very stable under these circumstances. This fact also makes the penalty formulation go through kinematic singular posi-

tions without problems, an advantage not shared by the classical Lagrange's method with Baumgarte stabilization (See Bayo and Avello (1993)).

The penalty formulation of Algorithm 5-3 will tend to be more efficient numerically than the Algorithms 5-4 and 5-5 in independent coordinates, because in Algorithm 5-3 the major computational burden is the formation, triangularization, and one forward reduction and backsubstitution of ($\mathbf{M} + \mathbf{\Phi}_\mathbf{q}^\mathrm{T} \boldsymbol{\alpha} \mathbf{\Phi}_\mathbf{q}$). Since the mass matrix does not modify the sparsity of the product ($\mathbf{\Phi}_\mathbf{q}^\mathrm{T} \mathbf{\Phi}_\mathbf{q}$), this operation is less costly than the formation, triangularization, and f forward reductions and backsubstitutions of ($\mathbf{\Phi}_\mathbf{q}^\mathrm{T} \mathbf{\Phi}_\mathbf{q}$) required for the formation of the matrix $\mathbf{R}$ in a single step of Algorithms 5-4 and 5-5. These algorithms also include the formation and triangularization of ($\mathbf{R}^\mathrm{T}\mathbf{M}\mathbf{R}$) which represents an additional computational burden of these methods.

5.3 Formulations Based on Velocity Transformations

In Section 5.2.3, a family of methods for transforming the dynamic equations from dependent to independent coordinates was presented. Equation (5.61), which defines the relation between dependent $\dot{\mathbf{q}}$ and independent velocities $\dot{\mathbf{z}}$, is really a particular case of the velocity transformation equations that can be introduced in the dynamic formulation. Equation (5.65) is the corresponding relation for accelerations. In this section, it will be shown how some velocity transformations can be used to improve the efficiency of the dynamic formulations described previously. These formulations, initially introduced by Jerkovsky (1978) and subsequently extended by other authors, such as: Kim and Vanderploeg (1986b), Nikravesh and Gim (1989), García de Jalón et al. (1990), and Bae and Won (1990) can be extremely efficient and simple to implement. However, they may have been presented in the literature in a rather involved way. We present these ideas in this section in a simple and yet rigorous manner, so that one can understand them easily. The concepts presented hereafter are independent of the coordinates used be they natural, reference points, or others. The efficiency of these formulations makes them be one of the best candidates for real time simulation. There will be a return to this topic in Chapter 8.

The numerical complexity in equation (5.67) comes from a double fact:

1) The computation of the matrix $\mathbf{R}$, that requires the factorization of the Jacobian matrix and as many forward and backward substitutions as columns that this matrix has; and,
2) The products of matrices that appear in equation (5.67), of which the most important and expensive to compute is the one on the left-hand side.

The relative importance of these computational tasks is problem dependent. Experience shows that very often each one of these two operations consisting of the computation of $\mathbf{R}$ and products of matrices requires around *40%* of the total computational cost involved in the numerical integration.

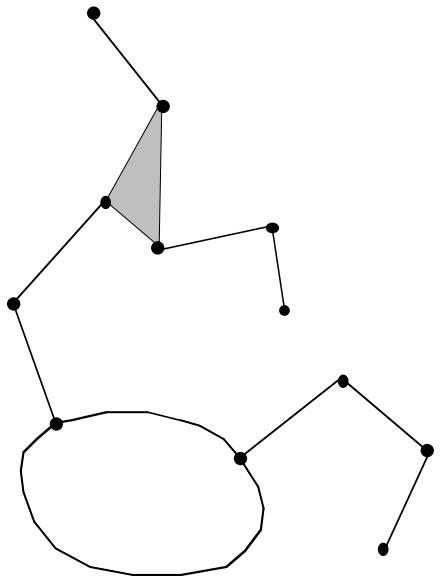

Figure 5.5. Tree-type planar multibody system.

The purpose of this section is to introduce velocity transformations similar to those of expressions (5.61) and (5.65) but with the difference of being particularly favorable in the sense of completely avoiding the Jacobian triangularization; hence allowing for an easy and efficient computation of the matrix **R** and term (**Sc**). It can be seen that these velocity transformations will not necessarily represent transformations between vectors of dependent and independent velocities but transformations between different or alternative sets of dependent velocities, that are particularly suitable from the point of view of improved numerical efficiency. Open- and closed-chain configurations are considered separately.

5.3.1 Open-Chain Multibody Systems

Multibody systems that have open kinematic chain or tree configuration are most appropriate to introduce the velocity transformations described in the previous section. In the sequel, the matrix **R** that relates the natural (or mixed) velocity vector $\dot{\mathbf{q}}$ and a set of independent velocities $\dot{\mathbf{z}}$ can be constructed directly with very few arithmetic operations and avoiding the formation and factorization of the Jacobian matrix $\mathbf{\Phi}_{\mathbf{q}}$.

An example of a tree-configured planar mechanism is presented in Figure 5.5. If the system has not gotten any fixed element, one of the elements of the system shall be defined as a *base body*. There is not a single choice for the base body, but in practice there are nearly always some natural or physical reasons

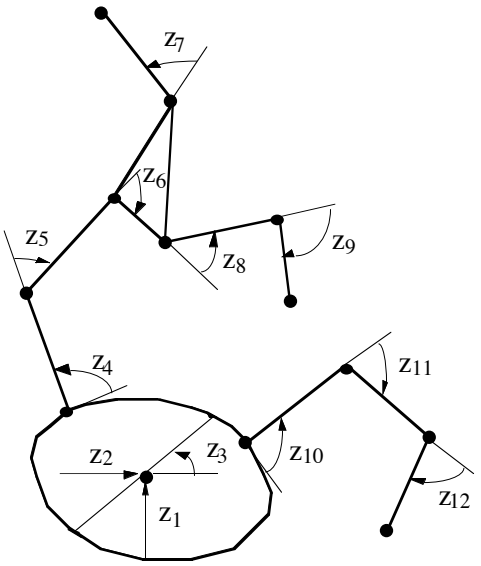

Figure 5.6. Numbering of the independent coordinates of a tree-type planar multibody system.

that determine the selection. The base body can have several branches departing from it, and some branches can also originate other branches.

An appropriate set of independent variables or coordinates for open-chain systems, such as the one shown in Figure 5.6, is determined by the variables that describe the rigid body motion of the base body plus the relative or joint coordinates that define the motion of each body with respect to the previous one in the corresponding branch of the chain. It is quite clear that this set of relative coordinates is independent. Figure 5.6 displays the base body plus relative coordinates of the planar system of Figure 5.5. In the example shown in Figures 5.5 and 5.6, it has been assumed that all the joints are of revolute type, although prismatic or any other joint type may be considered also.

There is a very easy way of constructing a matrix $\mathbf{R}$ for the system of Figure 5.6 without any need of forming and factoring the Jacobian matrix $\mathbf{\Phi_q}$. Remember that the columns of matrix $\mathbf{R}$ are a basis of the nullspace of the Jacobian matrix or, in other words, a base of the space of allowable velocities. This means that any velocity vector can be expressed as a linear combination of the columns of matrix $\mathbf{R}$. This is exactly what expression (5.61) represents. The column (i) of matrix $\mathbf{R}$ is the velocity vector in dependent coordinates $\dot{\mathbf{q}}$ obtained with:

$$\dot{z}_i = 1$$

$$\dot{z}_j = 0 \qquad j = 1, 2, \ldots n \ \ (j \neq i) \tag{5.71}$$

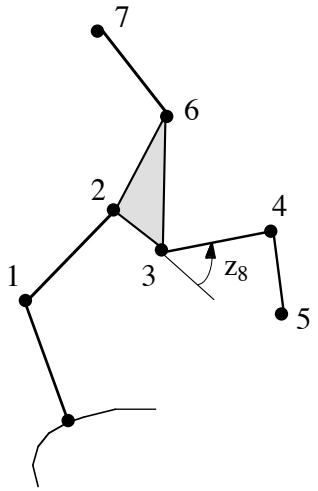

Figure 5.7. Detail numbering of one of the branches of the tree-type planar multibody system.

If the multibody system does not have closed loops, as is the case of the mechanism of Figure 5.6, this velocity vector; thus the corresponding column of matrix **R**, can be computed directly. In Figure 5.7, a branch of the mechanism of Figure 5.6 is displayed in more detail, including a possible numbering of some points.

Consider, for instance, the column of matrix **R** corresponding to the independent coordinate z_8. This column is obtained by giving a unit velocity to angle z_8, and no velocity to all the remaining coordinates z_j ($j \neq 8$). Only the column elements corresponding to points 4 and 5 will have a non-zero value. In general terms, only the dependent coordinates that are upwards in the branch of the independent coordinate being moved will introduce non-zero elements in the corresponding column of matrix **R**. It is very easy to take advantage of this well-defined sparsity structure on the computer implementation.

In the 3-D case, the concepts and computations are nearly as simple as in the planar case. The slightly higher complexity comes from: first, definition of the base body coordinates in 3-D; and second, the different kinds of joints that can appear in 3-D multibody systems. Only revolute, prismatic, cylindrical, universal, and spherical joints will be considered here.

5.3.1.1 Definition of Base Body Motion

There is no problem in finding six independent variables that define the velocity of the base body. Perhaps the simplest choice is determined by:

1) The three Cartesian components of the velocity of a reference point P; and,
2) The three Cartesian components of the base body angular velocity vector $\boldsymbol{\omega}$.

The difficulty arises because the velocities $\dot{\mathbf{z}}_b^T = \left\{ \mathbf{v}_P^T \; \boldsymbol{\omega}^T \right\}$ cannot be integrated to get the corresponding position variables due to the angular velocity part. There is no problem in using this base body velocity vector to compute the corresponding columns of matrix $\mathbf{R}$. The velocity vector $\dot{\mathbf{z}}_\mathbf{b}$ needs to be transformed into a different one, which can be called $\dot{\mathbf{z}}_\mathbf{B}$, and contains only integrable variables. The most important options are:

i) *Sets of three independent parameters, such as Euler or Bryant angles*. The limitation of this option is that none of these sets are free of singular positions; that is, positions for which the angular parameters are not determined unequivocally. If these positions can be effectively reached, it is necessary to foresee the cure, perhaps in the form of a change of reference frame.

ii) *Larger sets of dependent parameters, such as Euler parameters or quaternions*. Singularities can always be avoided, but there are constraint equations that relate the variables to be integrated. This dependency can be taken into account in the integration process. It is not really a very serious problem, as seen previously in this chapter.

It will always be possible to find a position-dependent matrix $\mathbf{W}$ that relates integrable and non-integrable base body velocities:

$$\dot{\mathbf{z}}_b = \mathbf{W}(\mathbf{z}_B)\dot{\mathbf{z}}_B \tag{5.72}$$

where $\dot{\mathbf{z}}_b$ is used to construct the matrix $\mathbf{R}$, and $\dot{\mathbf{z}}_B$ is used for the numerical integration process. This transformation shall be introduced in the vector of independent velocities $\dot{\mathbf{z}}$ before integrating it. It is not necessary to introduce it in $\ddot{\mathbf{z}}$ which can always be integrated once.

The first three columns of $\mathbf{R}$ related to the velocity of the reference point P can be computed as the result of applying three unit translations to the whole system on the inertial frame axes. In order to find a general expression, if $\dot{z}_1$, $\dot{z}_2$, and $\dot{z}_3$ are the related independent velocities, the velocity of a point *j*, and a unit vector $\mathbf{u}_j$, due to the translation of the base body can be expressed as:

$$\dot{\mathbf{r}}_j = \dot{z}_1 \, \mathbf{n}_1 + \dot{z}_2 \, \mathbf{n}_2 + \dot{z}_3 \, \mathbf{n}_3 \tag{5.73}$$

$$\dot{\mathbf{u}}_j = 0 \tag{5.74}$$

where ($\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$) are unit vectors on the inertial frame axes. From these equations, the elements of the columns of matrix $\mathbf{R}$, corresponding to point *j* and vector $\mathbf{u}_j$, can be computed in the form:

– column 1 $\quad \dot{z}_1 = 1 \quad \dot{z}_2 = 0 \quad \dot{z}_3 = 0$

– column 2 $\quad \dot{z}_1 = 0 \quad \dot{z}_2 = 1 \quad \dot{z}_3 = 0$

– column 3 $\quad \dot{z}_1 = 0 \quad \dot{z}_2 = 0 \quad \dot{z}_3 = 1$

If the independent velocities are the Cartesian components of the angular velocity vector $\boldsymbol{\omega}$, the columns 4 to 6 of the matrix $\mathbf{R}$ corresponding to the base body rotation may be computed as follows:

$$\dot{\mathbf{r}}_j = \dot{z}_4 \mathbf{n}_1 \wedge (\mathbf{r}_j - \mathbf{r}_P) + \dot{z}_5 \mathbf{n}_2 \wedge (\mathbf{r}_j - \mathbf{r}_P) + \dot{z}_6 \mathbf{n}_3 \wedge (\mathbf{r}_j - \mathbf{r}_P) \tag{5.75}$$

$$\dot{\mathbf{u}}_j = \dot{z}_4 \mathbf{n}_1 \wedge \mathbf{u}_j + \dot{z}_5 \mathbf{n}_2 \wedge \mathbf{u}_j + \dot{z}_6 \mathbf{n}_3 \wedge \mathbf{u}_j \tag{5.76}$$

From these formulas, columns 4 to 6 can be computed by making

– column 4 $\quad \dot{z}_4 = 1 \quad \dot{z}_5 = 0 \quad \dot{z}_6 = 0$

– column 5 $\quad \dot{z}_4 = 0 \quad \dot{z}_5 = 1 \quad \dot{z}_6 = 0$

– column 6 $\quad \dot{z}_4 = 0 \quad \dot{z}_5 = 0 \quad \dot{z}_6 = 1$

and this completes the information necessary to determine the part of $\mathbf{R}$ due to the base body degrees of freedom.

5.3.1.2 Different Joints in 3-D Multibody Systems

Figure 5.8 illustrates a generic joint *i*, a point *j*, and a unit vector $\mathbf{u}_j$ located upwards in the branch. The non-zero elements, corresponding to point *j* and vector $\mathbf{u}_j$ in the columns of matrix $\mathbf{R}$ associated with the degrees of freedom of joint *i*, are considered next.

Revolute joint. The joint variable is the angle z_i that defines the rotation of the joint around the axis determined by point *i* and vector $\mathbf{u}_i$. The velocity of point *j* induced by the relative velocity at joint *i* is

$$\dot{\mathbf{r}}_j = \dot{z}_i \mathbf{u}_i \wedge (\mathbf{r}_j - \mathbf{r}_i) \tag{5.77}$$

and the velocity induced in the unit vector $\mathbf{u}_j$:

$$\dot{\mathbf{u}}_j = \dot{z}_i \mathbf{u}_i \wedge \mathbf{u}_j \tag{5.78}$$

As done before, the corresponding values of the column (*i*) elements of the matrix $\mathbf{R}$ can be computed by giving a unit value to $\dot{\mathbf{z}}_i$.

Prismatic joint. Let z_i be the translational joint variable located on a line defined by point *i* and vector $\mathbf{u}_i$. The induced velocities of point *j* and vector $\mathbf{u}_j$ are:

$$\dot{\mathbf{r}}_j = \dot{z}_i \mathbf{u}_i \tag{5.79}$$

$$\dot{\mathbf{u}}_j = \mathbf{0} \tag{5.80}$$

These expressions allow for a very easy computation of the elements of the considered column of the matrix $\mathbf{R}$.

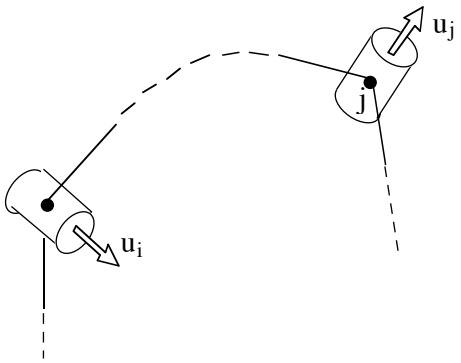

Figure 5.8. Description of a generic joint.

Cylindrical joint. This joint is different from the previous ones, since it has two degrees of freedom and it introduces two columns in matrix **R**. Let z_i and z_{i+1} be respectively the relative angle and distance that constitute the joint variables. Vector $\mathbf{u}_i$ determines the direction of both the rotation axis and the translation.

The velocities of point j and vector $\mathbf{u}_j$, due to the joint variables, are in this case:

$$\dot{\mathbf{r}}_j = \dot{z}_i \mathbf{u}_i \wedge (\mathbf{r}_j - \mathbf{r}_i) + \dot{z}_{i+1}\, \mathbf{u}_i \tag{5.81}$$

$$\dot{\mathbf{u}}_j = \dot{z}_i \mathbf{u}_i \wedge \mathbf{u}_j \tag{5.82}$$

The non-zero elements in the corresponding matrix **R** columns, can be obtained by making, respectively:

– column i $\quad \dot{z}_i = 1 \quad \dot{z}_{i+1} = 0$

– column $i+1$ $\quad \dot{z}_i = 0 \quad \dot{z}_{i+1} = 1$

Universal joint. This joint can be considered equivalent to two revolute joints, with the axes (belonging to different bodies) intersecting orthogonally at a common point. If $\dot{z}_i$ and $\dot{z}_{i+1}$ are the joint independent velocities, the elements related to point j and vector $\mathbf{u}_j$ in the columns of **R** come from the following velocity expressions:

$$\dot{\mathbf{r}}_j = \dot{z}_i\, \mathbf{u}_i \wedge (\mathbf{r}_j - \mathbf{r}_j) + \dot{z}_{i+1}\, \mathbf{v}_i \wedge (\mathbf{r}_j - \mathbf{r}_i) \tag{5.83}$$

$$\dot{\mathbf{u}}_j = \dot{z}_i\, \mathbf{u}_i \wedge \mathbf{u}_j + \dot{z}_{i+1}\, \mathbf{v}_i \wedge \mathbf{u}_j \tag{5.84}$$

From these expressions, the two columns of matrix **R** can be obtained by giving, respectively, the following values to the independent velocities:

– column i $\quad \dot{z}_i = 1 \quad \dot{z}_{i+1} = 0$

– column $i+1$ $\quad \dot{z}_i = 0 \quad \dot{z}_{i+1} = 1$

Spherical joint. The spherical joint allows for three rotations. This fact produces in the joint similar difficulties to the ones that were found at the time of defining the angular orientation of the base body. There is no problem in using three unit angular velocities on three orthogonal axes to compute the corresponding columns of matrix **R**. However, these independent velocities cannot be integrated to get displacement or position variables. It is necessary to transform those independent velocities into another set of integrable velocities that are dependent or independent according to an expression similar to (5.72).

If $\dot{z}_i^b$, $\dot{z}_{i+1}^b$ and $\dot{z}_{i+2}^b$ are the Cartesian components of the relative angular velocity vector, the velocities of point j and vector $\mathbf{u}_j$ are:

$$\dot{\mathbf{r}}_j = \dot{z}_i^b \ \mathbf{n}_1 \wedge (\mathbf{r}_j - \mathbf{r}_i) + \dot{z}_{i+1}^b \ \mathbf{n}_2 \wedge (\mathbf{r}_j - \mathbf{r}_i) + \dot{z}_{i+2}^b \ \mathbf{n}_3 \wedge (\mathbf{r}_j - \mathbf{r}_i) \qquad (5.85)$$

$$\dot{\mathbf{u}}_j = \dot{z}_i^b \ \mathbf{n}_1 \wedge \mathbf{u}_j + \dot{z}_{i+1}^b \ \mathbf{n}_2 \wedge \mathbf{u}_j + \dot{z}_{i+2}^b \ \mathbf{n}_3 \wedge \mathbf{u}_j \qquad (5.86)$$

where ($\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$) are orthogonal unit vectors on the inertial reference frame axes.

Using equations (5.85) and (5.86), the columns of matrix **R** (the terms corresponding to point j and vector $\mathbf{u}_j$) can be computed with the following values:

– column i $\quad \dot{z}_i^b = 1 \quad \dot{z}_{i+1}^b = 0 \quad \dot{z}_{i+2}^b = 0$

– column $i+1$ $\quad \dot{z}_i^b = 0 \quad \dot{z}_{i+1}^b = 1 \quad \dot{z}_{i+2}^b = 0$

– column $i+2$ $\quad \dot{z}_i^b = 0 \quad \dot{z}_{i+1}^b = 0 \quad \dot{z}_{i+2}^b = 1$

Once we have described how the columns of matrix **R** can be computed, the computation of the term (**Sc**) that appears in equation (5.67) remains. This term represents the dependent acceleration vector $\ddot{\mathbf{q}}$ computed with the true velocities $\dot{\mathbf{q}}$ or $\dot{\mathbf{z}}$ but with zero independent accelerations $\ddot{\mathbf{z}}$.

The elements of vector (**Sc**) corresponding to point j and vector $\mathbf{u}_j$ can be computed by adding to the acceleration of this point and vector the contribution of the true independent velocities of the base body and all the joints that are downwards in the branch of point j. The corresponding expressions are:

$$\ddot{\mathbf{r}}_j \big|_{\ddot{\mathbf{z}}=0} = \sum_k \ \dot{z}_k \ \left[\mathbf{u}_k \wedge (\dot{\mathbf{r}}_j - \dot{\mathbf{r}}_i) + \dot{\mathbf{u}}_k \wedge (\mathbf{r}_j - \mathbf{r}_i) \right] \qquad (5.87)$$

$$\ddot{\mathbf{u}}_j \big|_{\ddot{\mathbf{z}}=0} = \sum_k \ \dot{z}_k \ \left(\mathbf{u}_k \wedge \dot{\mathbf{u}}_j + \dot{\mathbf{u}}_k \wedge \mathbf{u}_i \right) \qquad (5.88)$$

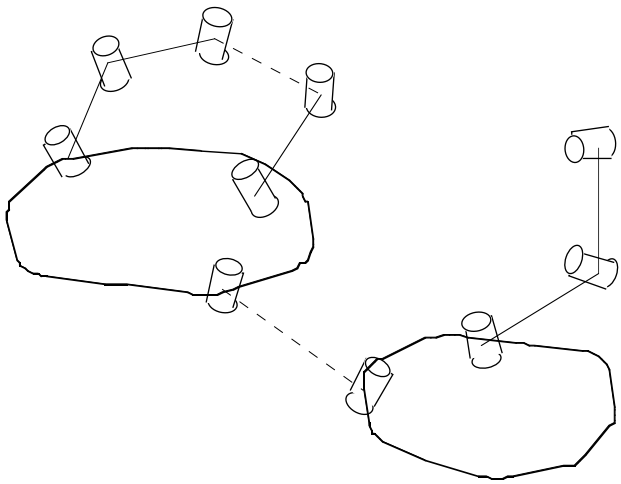

Figure 5.9. Closed-chain multibody system.

Hence, all the information necessary to compute the coefficients of equation (5.67) is available.

5.3.2 *Closed-Chain Multibody Systems*

In order to extend the previous formulation to multibody systems including closed loops, one should remember briefly the formulation of the dynamic equations using dependent coordinates and Lagrange multipliers through equation (5.10) repeated here:

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) \end{Bmatrix} \tag{5.89}$$

This equation describes the motion of the multibody system formulated with dependent coordinates. The natural coordinates can lead to a constant matrix $\mathbf{M}$, a Jacobian matrix $\boldsymbol{\Phi}_{\mathbf{q}}$ being a linear function of coordinates $\mathbf{q}$, and to no velocity-dependent inertia forces in the RHS of equation (5.89). The term $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})$ comes from the double differentiation of the constraint conditions and can be replaced by the term $\mathbf{g}$ (See equation (5.28)), if the Baumgarte constraint stabilization technique is desired.

After this brief introduction, the formulation described in the previous section can be extended to systems with closed loops and perhaps several base bodies, as in the system shown in Figure 5.9. Here, the system can be transformed into one or more open tree systems by opening the closed loops and disconnecting the base bodies. With natural coordinates, most of the constraint equations come from the rigid body conditions of the elements. The simplest way to transform

the system into one or more open trees is by removing the rigid body constraints corresponding to the closure of the loops and the connection between base bodies shown by dashed lines in the system of Figure 5.9. With reference point coordinates, constraints arise mainly from the system joints. In this case, the best way to open the loops is by removing some of them.

In order to obtain the dynamic equations, it is now very convenient to distinguish between the constraint equations corresponding to the open-chain systems (superscript 1) and the constraints corresponding to the closure conditions and connection between base bodies (superscript 2). Writing the dynamic equations using the Lagrange multiplier formulation of equation (5.89) and separating the two groups of constraints, we arrive at:

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{q}}^{1\mathrm{T}} & \boldsymbol{\Phi}_{\mathbf{q}}^{2\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}}^{1} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\Phi}_{\mathbf{q}}^{2} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda}^{1} \\ \boldsymbol{\lambda}^{2} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} \\ \mathbf{c}^{1} \\ \mathbf{c}^{2} \end{Bmatrix} \tag{5.90}$$

The velocity transformation corresponding to the open-chain constraints, whose Jacobian matrix nullspace is given by matrix $\mathbf{R}^1$ can now be introduced. Consequently,

$$\dot{\mathbf{q}} = \mathbf{R}^{1}\, \dot{\mathbf{z}} \tag{5.91}$$

and equation (5.90) becomes:

$$\begin{bmatrix} \mathbf{R}^{1\mathrm{T}}\mathbf{M}\,\mathbf{R}^{1} & \mathbf{R}^{1\mathrm{T}}\boldsymbol{\Phi}_{\mathbf{q}}^{2\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}}^{2}\,\mathbf{R}^{1} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{z}} \\ \boldsymbol{\lambda}^{2} \end{Bmatrix} = \begin{Bmatrix} \mathbf{R}_{1}^{\mathrm{T}}(\mathbf{Q} - \mathbf{M}\,\mathbf{S}^{1}\,\mathbf{c}^{1}) \\ \mathbf{c}^{2} - \boldsymbol{\Phi}_{\mathbf{q}}^{2}\,\mathbf{S}^{1}\,\mathbf{c}^{1} \end{Bmatrix} \tag{5.92}$$

The vector $\mathbf{z}$ in equations (5.91) and (5.92) is not a vector of independent coordinates as in Section 5.3.1, but dependent coordinates corresponding to the base bodies plus the relative coordinates of the open tree configuration joints. These coordinates are related by the constraint equations $\boldsymbol{\Phi}^2$. Then equation (5.91) is a transformation between two dependent velocity vectors. Vector $\dot{\mathbf{z}}$ will contain usually far less variables than vector $\dot{\mathbf{q}}$.

This formulation is also advantageous because the matrix $\mathbf{R}^1$ which corresponds to open-chain constraints can be constructed directly without any explicit Jacobian factorization as explained in Section 5.3.1. The term $(\boldsymbol{\Phi}_{\mathbf{q}}^{2}\,\mathbf{R}^{1})$ represents the projection of closure loop constraints (superscript 2) on the nullspace of the open-chain system (superscript 1).

Equation (5.92) has the form of the equations of motion in dependent coordinates with Lagrange multipliers to which Baumgarte stabilization can be directly applied. This could obviously be substituted by the penalty formulation (See Section 5.1.4), or by seeking a true independent set of coordinates. This can be done (See Section 5.2) by computing numerically the nullspace of the projected Jacobian matrix $(\boldsymbol{\Phi}_{\mathbf{q}}^{2}\,\mathbf{R}^{1})$ which can be a very small matrix.

5.4 Formulations Based on the Canonical Equations

Some authors have drawn attention recently to the use of the canonical equations as a way to improve numerically the formulation of the equations of motion of mechanical systems and to perform a faster and more stable simulation; thus more suitable for real time analysis. We discuss in this section the different approaches that can be derived from the use of the canonical equations for constrained mechanical systems, and whether the use of these equations may lead to more efficient and stable numerical implementations than those coming from acceleration-based formalisms. Specifically, we will describe the canonical equations that result: first, from the use of the Lagrange multipliers; secondly, from the use of independent coordinates; and thirdly, from the penalty formulation.

5.4.1 Lagrange Multiplier Formulation

Consider again a mechanical system whose configuration is characterized by a vector $\mathbf{q}$ of n generalized coordinates that are interrelated through the m kinematic constraint conditions $\mathbf{\Phi}(\mathbf{q}, t)=\mathbf{0}$, of the holonomic type. The Lagrange's equations of such a system are given in equation (5.2) which along with the constraint equations (5.1) constitute a set of *(n+m)* mixed differential algebraic equations (DAEs) of index three (See Chapter 7), with $\boldsymbol{\lambda}$ as the Lagrange multipliers. The *conjugate* or *canonical momenta* was defined in Chapter 4 as:

$$\mathbf{p} = \frac{\partial L}{\partial \dot{\mathbf{q}}} \tag{5.93}$$

along with the Hamiltonian:

$$H = \mathbf{p}^{\mathrm{T}}\, \dot{\mathbf{q}} - L \tag{5.94}$$

where the previously introduced matrix notation has been employed. Hamilton's equations for a constrained system are formulated (See Section 4.1.5) as:

$$\dot{\mathbf{q}} = \frac{\partial H}{\partial \mathbf{p}} \tag{5.95}$$

$$-\dot{\mathbf{p}} = \frac{\partial H}{\partial \mathbf{q}} - \mathbf{Q}_{\mathrm{ex}} + \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\, \boldsymbol{\lambda} \tag{5.96}$$

In the case of mechanical systems, the Lagrangian L is defined in terms of $\mathbf{q}$, $\dot{\mathbf{q}}$, and t and rather than following a lengthy process to form the Hamiltonian as an explicit function of $\mathbf{q}$, $\mathbf{p}$, and t, and then differentiating as in (5.95), the *canonical equations* can be directly obtained from (5.93) and (5.96). Since the system kinetic energy is a quadratic function of the generalized velocities, equations (5.93) and (5.96) directly lead to the following set of equations in matrix form:

$$\mathbf{p} = \mathbf{M}\,\dot{\mathbf{q}} \tag{5.97}$$

$$\dot{\mathbf{p}} = L_{\mathbf{q}} + \mathbf{Q}_{\text{ex}} - \boldsymbol{\Phi}_{\mathbf{q}}^{\text{T}}\,\boldsymbol{\lambda} \tag{5.98}$$

where $\mathbf{M}$ is the mass matrix, $L_{\mathbf{q}}$ is the partial derivative of the Lagrangian with respect to the coordinates, $\boldsymbol{\Phi}_{\mathbf{q}}$ is the Jacobian matrix of the constraint equations, and $\mathbf{Q}_{\text{ex}}$ is the vector of applied external and dissipative forces. The combination of equations (5.97)-(5.98) and the constraints conditions constitutes a system of *(2n+m)* differential and algebraic equations (DAE) of index two (See Chapter 7). Although there are *n* more equations than in equation (5.10), $\dot{\mathbf{p}}$ can be obtained explicitly by (5.98). In addition, index two DAEs are better behaved than index three DAEs (Brenan et al. (1989)). Therefore, the use of (5.97)-(5.98) may be numerically advantageous as compared to the use of (5.10), when using algorithms for the solution of the mixed differential algebraic equations.

In order to avoid the mixed differential and algebraic equations, Lankarani and Nikravesh (1988) modified the system Lagrangian to include the kinematic velocity constraints as:

$$L^{*} = L + \dot{\boldsymbol{\Phi}}^{\text{T}}\,\boldsymbol{\sigma} \tag{5.99}$$

where $\boldsymbol{\sigma}$ is a new set of Lagrange multipliers (It may be very easily demonstrated that $\boldsymbol{\lambda} = \dot{\boldsymbol{\sigma}}$.) The new Hamiltonian is $H^{*}=\mathbf{p}^{\text{T}}\dot{\mathbf{q}}-L^{*}$, and the application of (5.93) and (5.95) leads to:

$$\mathbf{p} = \mathbf{M}\,\dot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\text{T}}\,\boldsymbol{\sigma} \tag{5.100}$$

$$\dot{\mathbf{p}} = L_{\mathbf{q}} + \mathbf{Q}_{\text{ex}} + \dot{\boldsymbol{\Phi}}_{\mathbf{q}}^{\text{T}}\,\boldsymbol{\sigma} \tag{5.101}$$

That, along with

$$\dot{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_{\mathbf{q}}\,\dot{\mathbf{q}} \tag{5.102}$$

constitutes a set of $2n+m$ ordinary differential equations (ODE), with $\mathbf{p}$, $\mathbf{q}$, and $\boldsymbol{\sigma}$ as unknowns. If equation (5.100) is differentiated and substituted into the acceleration-based equation (5.4), the result is precisely the additional canonical equation (5.101). Thus, *the canonical equations originate from the acceleration-based equations by the mere canonical transformation* defined by equation (5.100).

Only *(n+m)* equations need to be solved at each time step in the numerical implementation of the algorithm, which can be described as follows:

Algorithm 5-6

1. Start at time *t* when $\mathbf{p}$ and $\mathbf{q}$ are known.
2. Use (5.100) along with (5.102) to solve for $\dot{\mathbf{q}}$ and $\boldsymbol{\sigma}$ at time *t*, as follows:

$$\begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{q}}^{\text{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & 0 \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}} \\ -\boldsymbol{\sigma} \end{Bmatrix} = \begin{Bmatrix} \mathbf{p} \\ 0 \end{Bmatrix} \tag{5.103}$$

3. Use (5.101) to compute $\dot{\mathbf{p}}$ explicitly, with no solution of equations involved.
4. Obtain the vectors **p** and **q** at time $(t+\Delta t)$ by the numerical integration:

$$\dot{\mathbf{s}}_t^{\mathrm{T}} \equiv \left\{ \dot{\mathbf{z}}^{\mathrm{T}},\ \dot{\mathbf{y}}^{\mathrm{T}} \right\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{s}_{t+\Delta t}^{\mathrm{T}} \equiv \left\{ \mathbf{z}^{\mathrm{T}},\ \mathbf{y}^{\mathrm{T}} \right\}_{t+\Delta t}$$

5. Upon convergence of the n.i.s., update the time variable and go to step 2.

Lankarani and Nikravesh (1988) showed in their numerical simulations that since only the first time derivative of the constraints is used, the integration of this equations is more efficient and more stable than the acceleration-based formulation. With the acceleration-based formulation to avoid the integration of the mixed differential algebraic equations, the constraint conditions need to be differentiated twice; thus leading to larger constraint violations.

5.4.2 Formulation Based on Independent Coordinates

The formulation of the canonical equations of motion can also be written as a function of a minimum set of independent coordinates. This is the approach followed by Bae and Won (1990) who used the velocity transformation method developed by Kim and Vanderploeg (1986) to transform the equations of motion from the Cartesian space to the joint space. They used an equivalence between the Lagrangian and Newton Euler formulation to derive the partial derivative of the kinetic energy with respect to the independent coordinates. We show in this section how the canonical equations in independent coordinates can be obtained very simply if one considers equations (5.97) and (5.98) as the starting point.

Given the constraint conditions (See equation (5.61)) $\mathbf{\Phi}(\mathbf{q}, t)=0$, one can find the two matrices **R** and **S** such that

$$\dot{\mathbf{q}} = \mathbf{R}\ \dot{\mathbf{z}} + \mathbf{S}\ \mathbf{b} \tag{5.104}$$

where $\dot{\mathbf{z}}$ represents a set of independent of velocities. The substitution of (5.104) into (5.97) yields

$$\mathbf{p} = \mathbf{M}\ \mathbf{R}\ \dot{\mathbf{z}}\ + \mathbf{M}\ \mathbf{S}\ \mathbf{b} \tag{5.105}$$

and pre-multiplying both sides of equations (5.105) and (5.98) by $\mathbf{R}^{\mathrm{T}}$ one can obtain, respectively,

$$\mathbf{R}^{\mathrm{T}}\ \mathbf{p} = \mathbf{R}^{\mathrm{T}}\ \mathbf{M}\ \mathbf{R}\ \dot{\mathbf{z}}\ + \mathbf{R}^{\mathrm{T}}\ \mathbf{M}\ \mathbf{S}\ \mathbf{b} \tag{5.106}$$

$$\mathbf{R}^{\mathrm{T}}\ \dot{\mathbf{p}} = \mathbf{R}^{\mathrm{T}}\ (L_{\mathbf{q}} + \mathbf{Q}_{\mathrm{ex}}) \tag{5.107}$$

where the term containing the Jacobian matrix has been dropped, since $\mathbf{R}^{\mathrm{T}}\mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}} = 0$ (See Chapter 3).

The new variable $\mathbf{y}=\mathbf{R}^{\mathrm{T}}\mathbf{p}$ can be defined as the projection of the canonical momenta over the subspace of allowable motions. Then

$$\dot{\mathbf{y}} = \dot{\mathbf{R}}^{\mathrm{T}}\ \mathbf{p} + \mathbf{R}^{\mathrm{T}}\ \dot{\mathbf{p}} \tag{5.108}$$

and the substitution of these two expressions into equations (5.106) and (5.107) leads to

$$\mathbf{y} = \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{R}\,\dot{\mathbf{z}} + \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{S}\,\mathbf{b} \tag{5.109}$$

$$\dot{\mathbf{y}} = \mathbf{R}^{\mathrm{T}}\,(L_{\mathbf{q}} + \mathbf{Q}_{\mathrm{ex}}) + \dot{\mathbf{R}}^{\mathrm{T}}\,\mathbf{p} \tag{5.110}$$

Substituting the value of $\mathbf{p}$ given by (5.105) into (5.110) one can arrive at the final set of canonical equations in independent coordinates:

$$\mathbf{y} = \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{R}\,\dot{\mathbf{z}} + \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{S}\,\mathbf{b} \tag{5.111}$$

$$\dot{\mathbf{y}} = \mathbf{R}^{\mathrm{T}}\,(L_{\mathbf{q}} + \mathbf{Q}_{\mathrm{ex}}) + \dot{\mathbf{R}}^{\mathrm{T}}\,(\mathbf{M}\,\mathbf{R}\,\dot{\mathbf{z}} + \mathbf{M}\,\mathbf{S}\,\mathbf{b}) \tag{5.112}$$

These two equations may be also obtained from the acceleration-based equations (5.67) by the mere canonical transformation of (5.111). Equations (5.111) and (5.112) constitute a set of *2(n–m)* first order ordinary differential equations. Since $\dot{\mathbf{y}}$ is given explicitly in (5.112), only *n–m* equations need to be solved for each function evaluation in the numerical implementation of the algorithm. This can be described as follows:

Algorithm 5-7

1. Start at time *t* in which $\mathbf{z}$ and $\mathbf{y}$ are known.
2. Solve the position and velocity problems to get $\mathbf{q}$ and $\dot{\mathbf{q}}$.
3. Obtain the matrices $\mathbf{R}$ and $\dot{\mathbf{R}}$.
4. Use (5.111) to solve for $\dot{\mathbf{z}}$. The solution of *n–m* equations is required.
5. Use (5.112) to solve for $\dot{\mathbf{y}}$ explicitly.
6. Obtain the vectors $\mathbf{z}$ and $\mathbf{y}$ at time $(t+\Delta t)$ by numerical integration:

$$\dot{\mathbf{s}}_t^{\mathrm{T}} \equiv \left\{\dot{\mathbf{z}}^{\mathrm{T}},\ \dot{\mathbf{y}}^{\mathrm{T}}\right\}_t \quad \xrightarrow{\text{n.i.s.}} \quad \mathbf{s}_{t+\Delta t}^{\mathrm{T}} \equiv \left\{\mathbf{z}^{\mathrm{T}},\ \mathbf{y}^{\mathrm{T}}\right\}_{t+\Delta t}$$

7. Upon convergence of the n.i.s., update the time variable and go to step 2.

This scheme can be compared to the *(n–m)* second order ordinary differential equations resulting from the acceleration-based formulation (equation (5.67)):

$$\mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{R}\,\ddot{\mathbf{z}} = \mathbf{R}^{\mathrm{T}}\,\mathbf{Q} + \mathbf{R}^{\mathrm{T}}\,\mathbf{M}\,\mathbf{S}\,[\,\dot{\mathbf{\Phi}}_t + \dot{\mathbf{\Phi}}_{\mathbf{q}}\,(\mathbf{R}\,\dot{\mathbf{z}} + \mathbf{S}\,\mathbf{b})] \tag{5.113}$$

where $\mathbf{Q} = \mathbf{Q}_{\mathrm{ex}} + L_{\mathbf{q}} - \dot{\mathbf{M}}\,\dot{\mathbf{q}}$ contains the external forces plus all the inertia terms coming from the differentiation of the Lagrangian. One can see that both methods require the triangularization of the same matrix ($\mathbf{R}^{\mathrm{T}}\mathbf{M}\,\mathbf{R}$) at each function evaluation. In addition, there might not be much advantage in using the canonical equations (5.111) and (5.112). Because although equation (5.113) involves more matrix manipulations than (5.111) and (5.112) and has a more complicated forcing term, the canonical approach requires the additional evaluation of $\dot{\mathbf{R}}$ with a sizable amount of computations.

5.4.3 Augmented Lagrangian Formulation in Canonical Form

We consider in this section the penalty augmented Lagrangian formulation in its canonical form. Its better accuracy and stability properties makes this method more attractive than the generic penalty formulation.

Basic Augmented Lagrangian Formulation. Equation (5.99) can be considered as the starting point to build a modified Lagrangian formulation that will not only contain the Lagrange multipliers $\boldsymbol{\sigma}$ but also the penalty terms of the previous section. Accordingly

$$L^* = L + \frac{1}{2}\dot{\boldsymbol{\Phi}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\dot{\boldsymbol{\Phi}} - \frac{1}{2}\boldsymbol{\Phi}^{\mathrm{T}}\,\boldsymbol{\Omega}^2\,\boldsymbol{\alpha}\,\boldsymbol{\Phi} + \dot{\boldsymbol{\Phi}}^{\mathrm{T}}\,\boldsymbol{\sigma}^* \tag{5.114}$$

In the limit when the constraint conditions are satisfied, the penalty terms vanish, and $\boldsymbol{\sigma}=\boldsymbol{\sigma}^*$. This is similar to the Lagrange's formulation $\dot{\boldsymbol{\sigma}}^*=\boldsymbol{\lambda}^*$ and after the augmented Lagrangian iteration when the constraints are satisfied to machine precision $\dot{\boldsymbol{\sigma}}=\boldsymbol{\lambda}$. The differentiation of L^* with respect to $\dot{\mathbf{q}}$ leads to the following new canonical momenta in matrix form:

$$\mathbf{p} = \frac{\partial L^*}{\partial \dot{\mathbf{q}}} = \mathbf{M}\dot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\dot{\boldsymbol{\Phi}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^* \tag{5.115}$$

The modified *Hamiltonian* can be written as $H^* = \mathbf{p}^{\mathrm{T}}\,\dot{\mathbf{q}} - L^*$ and the use of (5.96), including the dissipative Rayleigh forces of (5.30), leads to:

$$\left[\mathbf{M} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\boldsymbol{\Phi}_{\mathbf{q}}\right]\dot{\mathbf{q}} = \mathbf{p} - \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\boldsymbol{\Phi}_t + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^* \tag{5.116a}$$

$$\dot{\mathbf{p}} = \mathbf{Q} + L_{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\dot{\boldsymbol{\Phi}} - \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,(\boldsymbol{\Omega}^2\,\boldsymbol{\Phi} + 2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\dot{\boldsymbol{\Phi}}) + \dot{\boldsymbol{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^* \tag{5.116b}$$

Equations (5.116) constitute a set of $2n$ first order ordinary differential equations. However, $\dot{\mathbf{p}}$ is given in explicit form, and therefore only n algebraic equations need be solved at each function evaluation for the numerical implementation of the algorithm. The numerical simulations have shown that equations (5.116) tend to be numerically stiff due to all the penalty terms concentrated in the RHS of (5.116b). This numerical stiffness limits the possible choices of numerical integrators. Standard ODE integrators that are based on conditionally stable predictor-corrector multi-step formulae lead to an increased number of function evaluations. A modification of (5.116) is used in the next section that circumvents this problem.

Modified Augmented Lagrangian Formulation. The canonical equation (5.116a) may be also written as:

$$\mathbf{p} = \mathbf{M}\,\dot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\dot{\boldsymbol{\Phi}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^* \tag{5.117}$$

which indicates that the canonical momenta is stabilized through the addition of penalty terms that are proportional to the violation of the velocity constraint

equations. If equation (5.117) is differentiated and substituted into the acceleration-based augmented Lagrangian equation (5.48) the result is the additional canonical equation (5.116b).

However, we can achieve a better stabilization of the canonical momenta if we add to the RHS of (5.117) two additional penalty terms: one term proportional to the constraint violation and the other to its integral. Accordingly, we define a new momenta $\mathbf{p}$ as:

$$\mathbf{p} = \mathbf{M}\,\dot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\dot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\mu}\,\boldsymbol{\Omega}\,\boldsymbol{\Phi} + \boldsymbol{\Omega}^{2}\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau\right) - \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^{*} \tag{5.118}$$

By expanding the term $\dot{\boldsymbol{\Phi}}$, equation (32) becomes

$$\left(\mathbf{M} + \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\,\boldsymbol{\Phi}_{\mathbf{q}}\right)\dot{\mathbf{q}} = \mathbf{p} - \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\boldsymbol{\Phi}_{t} + 2\,\boldsymbol{\mu}\,\boldsymbol{\Omega}\,\boldsymbol{\Phi} + \boldsymbol{\Omega}^{2}\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau\right) - \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^{*} \tag{5.119a}$$

The differentiation of (5.119a) and substitution into (5.48) leads to the second set of modified canonical equations:

$$\dot{\mathbf{p}} = \mathbf{Q} + L_{\mathbf{q}} + \dot{\boldsymbol{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\alpha}\left(\dot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\Omega}\,\boldsymbol{\mu}\,\boldsymbol{\Phi} + \boldsymbol{\Omega}^{2}\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau\right) + \dot{\boldsymbol{\Phi}}_{\mathbf{q}}^{\mathrm{T}}\,\boldsymbol{\sigma}^{*} \tag{5.119b}$$

which along with (5.119a) constitute a set of *2n* first order ordinary differential equations in the unknowns $\mathbf{p}$, $\mathbf{q}$, and $\boldsymbol{\sigma}^{*}$. Again, only *n* algebraic equations need be solved at each function evaluation for the numerical implementation of the algorithm. Contrary to equations (5.116), equations (5.119) do not become stiff. They even provide more numerical accuracy and better constraint stabilization than the acceleration-based formulation of equation (5.48).

We can compare this set of equations with the *n* second order ordinary differential equations resulting from the acceleration-based formulation. While both formulations require the triangularization of the same leading matrix for each function evaluation, there are advantages in the use of (5.119) as compared to (5.48). The kinematic constraint conditions are differentiated only once with the canonical procedure and twice, in the acceleration based formulation. This will lead to less violations of the constraints. It is shown in Bayo and Avello (1993) how this factor becomes detrimental for the acceleration-based formulation under repetitive singular positions, whereas the canonical approach leads to a much better performance.

The multipliers $\boldsymbol{\sigma}^{*}$ do not need to be solved explicitly. Following the same procedure as that used with the acceleration-based augmented Lagrangian formulation, the $\boldsymbol{\sigma}^{*}$ may be obtained in an iterative manner as:

$$\boldsymbol{\sigma}_{\mathrm{i+1}}^{*} = \boldsymbol{\sigma}_{\mathrm{i}}^{*} + \left(\dot{\boldsymbol{\Phi}} + 2\,\boldsymbol{\mu}\,\boldsymbol{\Omega}\,\boldsymbol{\Phi} + \boldsymbol{\Omega}^{2}\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau\right)_{\mathrm{i+1}} \qquad i = 0,\,1,\,2,\,\ldots \tag{5.120}$$

with $\boldsymbol{\sigma}_0^* = 0$ for the first iteration. Equation (5.119a), including the iterative process of (5.120), becomes

$$\left(\mathbf{M} + \boldsymbol{\Phi}_\mathbf{q}^\mathrm{T}\,\boldsymbol{\alpha}\,\boldsymbol{\Phi}_\mathbf{q}\right)\dot{\mathbf{q}}_{i+1} = \mathbf{M}\,\dot{\mathbf{q}}_i - \boldsymbol{\Phi}_\mathbf{q}^\mathrm{T}\,\boldsymbol{\alpha}\,(\boldsymbol{\Phi}_t + 2\,\boldsymbol{\mu}\,\boldsymbol{\Omega}\,\boldsymbol{\Phi} + \boldsymbol{\Omega}^2\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau) \qquad i = 0, 1, 2, \ldots \tag{5.121}$$

with $\mathbf{M}\dot{\mathbf{q}}_o = \mathbf{p}$ for the first iteration. Equation (5.121) shows that the velocity calculation at each function evaluation is refined so that the weighted summation of the constraint equations (5.120) is satisfied to machine precision. After the velocity calculation equation, (5.119b) may be used to evaluate the derivative of the canonical momenta.

Algorithm 5-8

1. Start at time t in which $\mathbf{p}$ and $\mathbf{q}$ are known.
2. Use (5.121) iteratively to solve for $\dot{\mathbf{q}}$, with $\mathbf{M}\,\dot{\mathbf{q}}_o = \mathbf{p}$ for the first iteration. At the end of each iteration use (5.120) to calculate the Lagrange multipliers $\boldsymbol{\sigma}^*$.
3. Use (5.119b) to compute $\dot{\mathbf{p}}$ explicitly with no solution of equations involved.
4. Call the numerical integration subroutine to compute $\mathbf{p}$ and $\mathbf{q}$ at time $t+\Delta t$.
5. Upon convergence of the n.i.s., if desired, use a differentiation scheme to obtain $\boldsymbol{\lambda} = \dot{\boldsymbol{\sigma}}$.
6. Update the time variable and go to step 2.

This algorithm is as efficient numerically as Algorithm 5-3, but much more stable under repetitive singular positions.

Canonical Augmented Lagrangian Formulation for Non-Holonomic Systems. The modified augmented Lagrangian formulation described above may also be extended to non-holonomic systems with constraints of the form:

$$\boldsymbol{\Phi}(\dot{\mathbf{q}}, \mathbf{q}, t) = \mathbf{0} \tag{5.122}$$

Typically, non-holonomic constraint conditions for multibody systems are such, that

$$\boldsymbol{\Phi} = \mathbf{A}(\mathbf{q}, t)\,\dot{\mathbf{q}} + \mathbf{B}(\mathbf{q}, t) \tag{5.123}$$

The acceleration-based augmented Lagrangian formulation for this type of constraints is

$$\mathbf{M}\,\ddot{\mathbf{q}} = \mathbf{Q} + L_\mathbf{q} - \dot{\mathbf{M}}\,\dot{\mathbf{q}} - \mathbf{A}^\mathrm{T}\,\boldsymbol{\alpha}\left(\dot{\boldsymbol{\Phi}} + \boldsymbol{\mu}\,\boldsymbol{\Phi}\right) - \mathbf{A}^\mathrm{T}\,\boldsymbol{\lambda}^* \tag{5.124}$$

In order to obtain the canonical equations, we follow a procedure similar to that used for the holonomic case and establish the following canonical transformation

$$\mathbf{p} = \mathbf{M}\,\dot{\mathbf{q}} + \mathbf{A}^\mathrm{T}\,\boldsymbol{\alpha}\,(\boldsymbol{\Phi} + \boldsymbol{\mu}\int_{t_o}^{t}\boldsymbol{\Phi}\,d\tau) + \mathbf{A}^\mathrm{T}\,\boldsymbol{\sigma}^* \tag{5.125a}$$

which indicates that a better stabilization of the canonical momenta may be achieved by considering one penalty term proportional to the constraint violation and the other term proportional to its integral. The differentiation of (5.125a) and posterior substitution into (5.124) leads to the second set of canonical equations:

$$\dot{\mathbf{p}} = \mathbf{Q} + L_{\mathbf{q}} + \dot{\mathbf{A}}^{\mathrm{T}} \boldsymbol{\alpha} (\boldsymbol{\Phi} + \boldsymbol{\mu} \int_{t_o}^{t} \boldsymbol{\Phi}\, d\tau) + \dot{\mathbf{A}}^{\mathrm{T}} \boldsymbol{\sigma}^{*} \qquad (5.125b)$$

which along with (5.125a) constitutes a set of *2n* first order ordinary differential equations in the unknowns $\mathbf{p}$, $\mathbf{q}$, and $\boldsymbol{\sigma}^{*}$. Again only *n* equations need to be solved at each function evaluation. The multipliers are given by

$$\boldsymbol{\sigma}^{*}_{\mathrm{i+1}} = \boldsymbol{\sigma}^{*}_{\mathrm{i}} + (\boldsymbol{\Phi} + \boldsymbol{\mu} \int_{t_o}^{t} \boldsymbol{\Phi}\, d\tau)_{\mathrm{i+1}}, \quad i = 0, 1, 2, \ldots \qquad (5.126)$$

with $\boldsymbol{\sigma}^{*}_{0} = 0$ for the first iteration.

References

Bae, D.S. and Won, Y.S., "A Hamiltonian Equation of Motion for Real-time Vehicle Simulation", *Advances in Design and Automation 1990*, Vol. 2, pp. 151-157, ASME Press, (1990).

Bae, D.S. and Yang, S.M., "A Stabilization Method for Kinematic and Kinetic Constraint Equations", *Real-Time Integration Methods for Mechanical System Simulation*, NATO ASI Series, Vol. 69, pp. 209-232, Springer-Verlag, (1990).

Baumgarte, J., "Stabilization of Constraints and Integrals of Motion in Dynamical Systems", *Computer Methods in Applied Mechanics and Engineering*, Vol. 1, pp. 1-16, (1972).

Bayo, E., García de Jalón, J., and Serna, M.A., "A Modified Lagrangian Formulation for the Dynamic Analysis of Constrained Mechanical Systems", *Computer Methods in Applied Mechanics and Engineering*, Vol. 71, pp. 183-195, (1988).

Bayo, E., García de Jalón, J., Avello, A., and Cuadrado, J., "An Efficient Computational Method for Real Time Multibody Dynamic Simulation in Fully Cartesian Coordinates", *Computer Methods in Applied Mechanics and Engineering*, Vol. 92, pp. 377-395, (1991).

Bayo, E. and Avello, A., "Singularity Free Augmented Lagrangian Algorithms for Constraint Multibody Dynamics", to appear in the *Journal of Nonlinear Dynamics*, (1993).

Brenan, K.E, Campbell, S.L., and Petzold, L.R., *The Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, Elsevier Science Publishing, (1989).

Chang, C.O. and Nikravesh, P.E., "An Adaptive Constraint Violation Stabilization Method for Dynamic Analysis of Mechanical Systems", *ASME Journal of Mechanisms, Transmissions and Automation in Design*, Vol. 107, pp. 488-492, (1985).

García de Jalón, J., Avello, A., Jiménez, J.M., Martín, F., and Cuadrado, J., "Real Time Simulation of Complex 3-D Multibody Systems with Realistic Graphics", *Real-Time Integration Methods for Mechanical System Simulation*, NATO ASI Series, Vol. 69, pp. 265-292, Springer-Verlag, (1990).

Gear, C.W., *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, (1971).

Goldstein H., *Classical Mechanics*, 2nd edition, Addison-Wesley, (1980).

Haug, E.J., *Computer–Aided Kinematics and Dynamics of Mechanical Systems, Volume I: Basic Methods*, Allyn and Bacon, (1989).

Jerkovsky, W., "The Structure of Multibody Dynamic Equations", *Journal of Guidance and Control*, Vol. 1, pp. 173-182, (1978).

Kamman, J.W. and Huston, R.L., "Dynamics of Constrained Multibody Systems", *ASME Journal of Applied Mechanics*, Vol. 51, pp. 899-903, (1984).

Kim, S.S. and Vanderploeg, M.J., "A General and Efficient Method for Dynamic Analysis of Mechanical Systems Using Velocity Transformations", *ASME Journal of Mechanisms, Transmissions and Automation in Design*, Vol. 108, pp. 176-182, (1986a).

Kim, S.S. and Vanderploeg, M.J., "QR Decomposition for State Space Representation of Constrained Mechanical Dynamic Systems", *ASME Journal on Mechanisms, Transmissions and Automation in Design,* Vol. 108, pp. 183-188, (1986b).

Kurdila, A. J. and Narcowich F.J., "Sufficient Conditions for Penalty Formulation Methods in Analytical Dynamics", to appear in *Computational Mechanics,* (1993).

Lankarani, H.M. and Nikravesh, P.E., "Application of the Canonical Equations of Motion in Problems of Constrained Multibody Systems with Intermittent Motion", *Advances in Design Automation 1988*, DE-Vol. 14, pp. 417-423, edited by S.S. Rao, ASME Press, (1988).

Mani, N.K., Haug, E.J., and Atkinson, K.E., "Application of Singular Value Decomposition for Analysis of Mechanical System Dynamics", *ASME Journal on Mechanisms, Transmissions and Automation in Design*, Vol. 107, pp. 82-87, (1985).

Nikravesh, P.E., "Some Methods for Dynamic Analysis of Constrained Mechanical Systems: A Survey", *Computer-Aided Analysis and Optimization of Mechanical System Dynamics*, ed. by E.J. Haug, Springer-Verlag, pp. 351-368, (1984).

Nikravesh, P.E. and Gim, G., "Systematic Construction of the Equations of Motion for Multibody Systems Containing Closed Kinematic Loops", *Advances in Design Automation 1989*, Vol. 3, pp. 27-33, ASME Press, (1989).

Oden, J.T., *Finite Elements. A Second Course, Volume II*, Chapter 3, Prentice-Hall, (1983).

Park, T.W. and Haug, E.J., "A Hybrid Numerical Integration Method for Machine Dynamic Simulation", *ASME Journal of Mechanisms, Transmissions and Automation in Design*, Vol. 108, pp. 211-216, (1986).

Pars, L.A., *A Treatise of Analytical Dynamics,* William Heineman Ltd., (1965).

Paul, B., "Analytical Dynamics of Mechanisms - A Computer-Oriented Overview", *Mechanism and Machine Theory*, Vol. 10, pp. 481-507, (1975).

Serna, M.A., Avilés, R., and García de Jalón, J., "Dynamic Analysis of Planar Mechanisms with Lower-Pairs in Basic Coordinates", *Mechanism and Machine Theory*, Vol. 17, pp. 397-403, (1982).

Shampine, L. and Gordon, M., *Computer Solution of ODE. The Initial Value Problem*, Freeman, (1975).

Steigerwald, M.F., "BDF Methods for DAEs in Multibody Dynamics: Shortcomings and Improvements", in *Real-Time Integration Methods for Mechanical System Simulation*, NATO ASI Series, Vol. 69, pp. 345-352, Springer-Verlag, (1990).

Unda, J., García de Jalón, J., Losantos, F., and Enparantza, R., "A Comparative Study of Some Different Formulations of the Dynamic Equations of Constrained Mechanical Systems", *ASME Journal of Mechanisms, Transmissions and Automation in Design*, Vol. 109, pp. 466-474, (1987).

Vanderplaats, G.N., *Numerical Optimization Techniques for Engineering Design: with Applications*, McGraw-Hill, (1984).

Wehage, R.A. and Haug, E.J., "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems", *ASME Journal of Mechanical Design*, Vol. 104, pp. 247-255, (1982).

Problems

5/1 Using natural coordinates, write the equations of motion of the slider-crank mechanism of the figure with: a) Lagrange multipliers b) Penalty formulation. Assume that the mass is uniformly distributed and the center is located at the middle of each element. Also $L_2=L_3/2=L$, $m_2=1$, $m_3=2$, and $m_4=1$.

5/2 Form the matrix $\mathbf{R}$ of the mechanism of Problem 5/1 and find the equations of motion: a) in dependent coordinates using $\mathbf{q}^T=(x_1,y_1,x_2)$ and equation (5.17); and b) in independent coordinates using $z_1=x_2$ and equation (5.67). For case a) discuss the regularization process to avoid the singular position when $L_3=L_2$ and both slider and crank are in the vertical position.

5/3 Repeat Problem 5/2 using the canonical formulation (equations (5.111) and (5.112)) in independent coordinates using $z_1=x_2$.

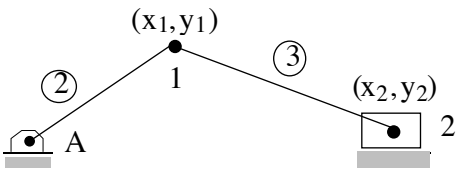

Figure P5/1.

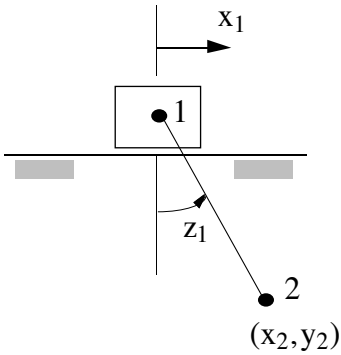

Figure P5/4.

5/4 Find the equations of motion of the mechanism of the figure when the coordinate x_1 is kinematically imposed. Use: a) Dependent coordinates with Lagrange multipliers; b) independent coordinates choosing the angle z_1 as the independent variable.

5/5 Solve Problem 5/3 using the canonical formulation of Section 5.4 with dependent coordinates and the penalty formulation for the constraint equations.

5/6 Solve for the equations of motion of the mechanism shown in the figure, using the velocity transformation methods of Section 5.3. Open the loop at joint 2 where indicated and then apply the closure conditions using the Lagrange multiplier approach with Baumgarte stabilization.

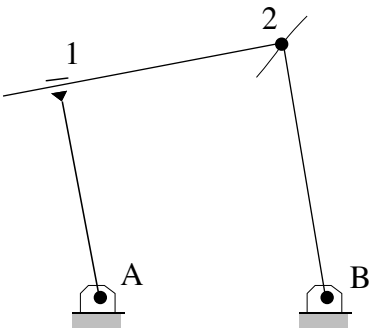

Figure P5/6.

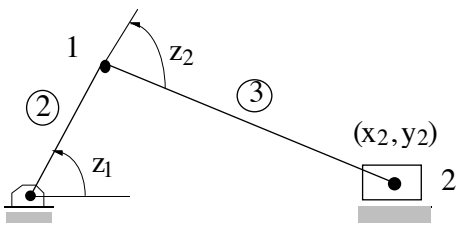

Figure P5/7.

5/7 Form the differential equations of motion of the mechanism of the figure, using velocity transformations and the penalty formulation for the closure condition $y_2=0$.

5/8 Derive the equations of motion of a coin that rolls over a flat surface with no slipping, using the natural coordinates that are shown in the figure. The non-holonomic constraint condition is given by $\mathbf{v}_P=0$ (3 equations).

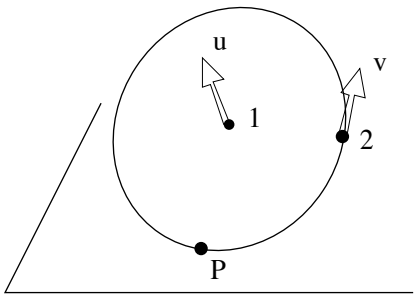

Figure P5/8.

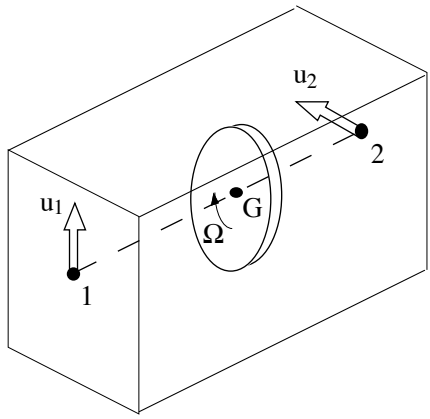

Figure P5/9.

5/9 Consider a satellite that is modeled with two points 1 and 2, and two unit vectors $\mathbf{u}_1$ and $\mathbf{u}_2$. This satellite contains a high speed rotor that rotates at a constant relative angular velocity Ω. Model the effect of the rotor by means of an equivalent set of forces.

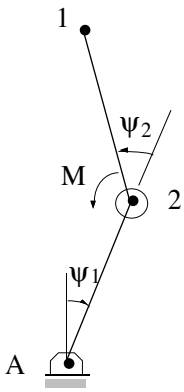

Figure P5/10.

5/10 The controlled mechanical system in the figure consists of two rods with two revolute joints that move on a vertical plane. Joint A is torque free. Joint 2 has an actuator that applies two opposite torques on both bars, so as to keep the system in the vertical position ($Y_1 = Y_2 = 0$). Write the differential equations of motion using Y_1 and Y_2 as independent coordinates.