A Constraint Force Algorithm for Formulating Equations of Motion

Paul C. Mitiguy[*] Arun K. Banerjee[†]

April 12, 2000

Abstract

Constraint force algorithms for static and dynamic analysis allow for a tremendous amount of flexibility in motion analysis. This algorithm, partly developed by dynamicists at MSC.Software, is referred to as the *Constraint Force Algorithm*, and has several advantages over other popular methods for formulating equations of motion. For example,

- Constraint force algorithms simulate systems with varying degrees of freedom.
- Constraint force algorithms simulate systems with static and kinetic Coulomb Friction.
- Constraint force algorithms handle collisions in a seamless fashion.
- Constraint force algorithms allow the user to specify various motions through length, velocity, and acceleration actuators.
- Constraint force algorithms deal with inequality constraints such as rope and separators.

The *Constraint Force Algorithm* is a variation of the well known Newton-Euler equations of motion, which can be written[1] as

$$\begin{aligned}\mathbf{F} &= m\mathbf{a} \\ \mathbf{T} &= \underline{\mathbf{I}}\cdot\boldsymbol{\alpha} + \boldsymbol{\omega}\times\underline{\mathbf{I}}\cdot\boldsymbol{\omega}\end{aligned}$$

One problem with the Newton-Euler "free-body" method is that it leads to a large set of equations whose unknowns are constraint forces *and* accelerations. To avoid the prohibitively vast amount of time it takes to solve large sets of linear equations at each integration step, the constraint force algorithm breaks the problem into two parts. First, it uses linear algebra techniques to isolate and solve a relatively small set of equations for the constraint forces only. Then, with all the forces on each body known, it solves for the accelerations, without solving *any* additional linear equations. This leads to huge computational savings because the number of operations required to solve a set of linear equations varies with the number of equations *cubed*. In light of these facts, it is not surprising that constraint force algorithms, such as those employed by VISUAL NASTRAN$^{\text{TM}}$, run nearly *8 times* faster than dynamics programs that use the Newton-Euler algorithm.

[*]Principal Technical Developer, MSC.Software, 66 Bovet Rd. San Mateo CA 94402

[†]Consulting Engineer, 92-30/250 Lockheed Martin Advanced Technology Center, Palo Alto, CA 94304

[1]For certain planar systems, Euler's equation of motion can be simplified to $\mathbf{T} = I\,\boldsymbol{\alpha}$

Constraint Forces and Applied Forces

One idea, central to the Constraint Force Algorithm, is to distinguish between *applied forces* and *constraint forces*. *Applied forces* are those forces whose magnitude and line of action are known by the analyst without reliance on a dynamic principle. As such, it is usually possible to write an explicit equation for an applied force. Forces of this type include gravitational, electrical, magnetic, kinetic friction, spring, damper, and forces exerted by control-systems. *Constraint forces* are those forces which are not applied forces, meaning that their magnitude and/or line of action must be determined through application of a dynamic principle. The constraint forces are usually determined by the type of joint, (e.g., pin joint, rod, slot, pulley, etc.,) connecting the bodies.

Motion Variables

Motion variables are quantities which describe the translational and/or rotational speed of objects in a physical system. They are usually defined as linear combinations of time derivatives of configuration variables (quantities which describe the position and/or orientation of system components). Recently, there has been a substantial amount of interest in selecting motion variables which minimize numerical simulation time. One choice of motion variables which works well for analyzing two-dimensional motions of rigid bodies is the variables v_x, v_y, and ω, where v_x and v_y measure the inertial velocity of the mass center of each body, and ω measures the inertial angular speed of each body.

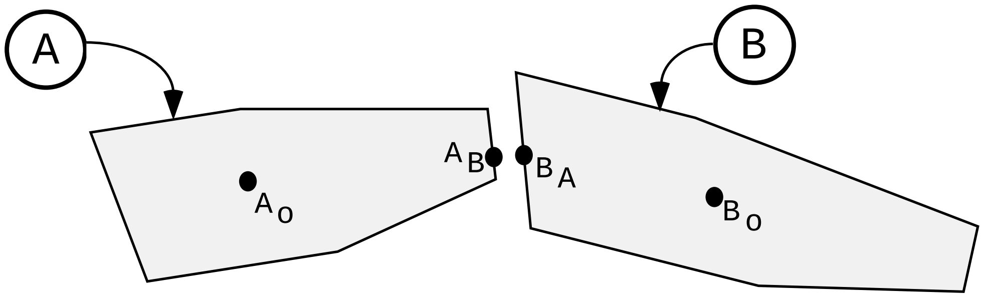

Figure 1: Two Bodies Connected by a Pin Joint

Conceptual Example

The constraint force algorithm can be demonstrated most simply with a two-dimensional example. The system depicted in Figure 1 consists of a body A connected to a body B by a revolute joint at point A_B of A and point B_A of B.

Constraint Equations

To account for the fact that the motions of A and B are not independent, one can write constraint equations which arise from the fact that the accelerations of A_B and B_A are equal. These constraint equations, written in terms of the `2x3` matrices G_A and G_B, the `3x1` matrices $\dot{U}_A$ and $\dot{U}_B$ defined as

$$\dot{U}_A = \begin{bmatrix} \dot{v}_x^{A_o} & \dot{v}_y^{A_o} & \dot{\omega}^A \end{bmatrix}^T \quad (1)$$

$$\dot{U}_B = \begin{bmatrix} \dot{v}_x^{B_o} & \dot{v}_y^{B_o} & \dot{\omega}^B \end{bmatrix}^T \quad (2)$$

and the `2x1` matrix C, are

$$G_A\,\dot{U}_A \;+\; G_B\,\dot{U}_B = C \tag{3}$$

Specifically, equation (3) arises from setting $\mathbf{a}^{A_B}$, the inertial acceleration of A_B, equal to $\mathbf{a}^{B_A}$, the inertial acceleration of B_A, where

$$\begin{aligned}
\mathbf{a}^{A_B} &= \mathbf{a}^{A_o} + \boldsymbol{\alpha}^A \times \mathbf{p}^{A_B/A_o} + \boldsymbol{\omega}^A \times (\boldsymbol{\omega}^A \times \mathbf{p}^{A_B/A_o}) \\
\mathbf{a}^{B_A} &= \mathbf{a}^{B_o} + \boldsymbol{\alpha}^B \times \mathbf{p}^{B_A/B_o} + \boldsymbol{\omega}^B \times (\boldsymbol{\omega}^B \times \mathbf{p}^{B_A/B_o})
\end{aligned}$$

Laws of Motion

The equations governing the motion of A may be written in terms of M_A, a `3x3` matrix whose elements are the mass and central moment of inertia of A; F, a 2x1 matrix of constraint actions, given by

$$F = [F_x \;\; F_y]^T \tag{4}$$

R_A, a 3x1 matrix whose elements are related to the applied forces on A; G_A^T, the transpose of G_A; and $\dot{U}_A$. The equations governing the motion of B may be written in a similar fashion so that the two sets of equations can be cast as

$$M_A\,\dot{U}_A \;+\; G_A^T\,F \;=\; R_A \tag{5}$$

$$M_B\,\dot{U}_B \;+\; G_B^T\,F \;=\; R_B \tag{6}$$

The solution process for $\dot{U}_A$, $\dot{U}_B$, and F begins by forming some intermediate matrices X_A, X_B, Y_A, and Y_B which can be written in terms of M_A^{-1} and M_B^{-1} the matrix inverses[2] of M_A and M_B,

$$X_A = M_A^{-1}\,G_A^T \tag{7}$$

$$Y_A = M_A^{-1}\,R_A \tag{8}$$

$$X_B = M_B^{-1}\,G_B^T \tag{9}$$

$$Y_B = M_B^{-1}\,R_B \tag{10}$$

Solving equation (5) for $\dot{U}_A$, and subsequent substitution of equation (7) and equation (8), and proceeding similarly for $\dot{U}_B$, leads to

$$\dot{U}_A = Y_A - X_A\,F \tag{11}$$

$$\dot{U}_B = Y_B - X_B\,F \tag{12}$$

Substitution of equations (11) and (12) into equation (3) leads to

$$(G_A\,X_A + G_B\,X_B)\;F = G_A\,Y_A \;+\; G_B\,Y_B - C \tag{13}$$

The solution for F, the constraint actions, is found by solving the 2x2 system of equations in equation (13), and The solution for $\dot{U}_A$ is found by substituting the now known values of F into equation (11). Similarly, $\dot{U}_B$ is found by substituting F into equation (12).

[2]It is usually numerically disadvantageous to explicitly form M_A^{-1}. One may instead solve the sets of linear equations $[M_A][X_A|Y_A] = [G_A^T|R_A]$ for X_A and Y_A.