Disclaimer of All Warranties & Liabilities

Engineering Dynamics Corporation (EDC) makes no warranties, either express or implied, with respect to this manual, or with respect to the software described in this manual, its quality, performance, merchantability, or fitness for any particular purpose. EDC software is sold or licensed "AS IS". The entire risk as to its quality and performance is with the buyer. Should the programs prove defective following their purchase, the buyer (and not EDC) assumes the entire cost of all necessary servicing, repair, or correction and any incidental or consequential damages. In no event will EDC be liable for direct, indirect, incidental, or consequential damages resulting from any defect in the software, even if EDC has been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the limitation or exclusion may not apply to you.

Proper use of the accident reconstruction software described herein requires a thorough understanding of vehicle dynamics. Therefore, you must agree to assume full responsibility for any decisions which are based, in whole or in part, upon information obtained by using this software. EDC does not warrant that the functions contained in the program will meet your requirements or that the operation of the program will be uninterrupted or error free.

BY USING THIS PROGRAM, YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS.

Trademarks

EDVAP™, EDSVS™, EDVTS™, EDSMAC™, EDCAD™, and RTG™ are trademarks of Engineering Dynamics Corporation. IBM® is a registered trademark of International Business Machines Corp.

CONTENTS

FIGURES

INTRODUCTION

EDC simulation programs are used for the computer simulation of motor vehicle accidents. This manual describes the calculation procedures used by three simulation programs

- EDSVS (Single Vehicle Simulator)
- EDVTS (Vehicle-Trailer Simulator)
- EDSMAC (Two Vehicle Accident Simulator)

The purpose of this publication is to provide the user of these EDC simulation programs with an explanation of how they work. The technical portion is divided into five major chapters:

Anatomy of a Simulation describes the basic program elements common to all EDC simulation programs. An overview of each program, including flow charts, is presented.

Vehicle Models describes the mathematical/physical model used by each program. A complete free-body analysis is presented.

Tire Forces begins with a general overview of basic tire mechanics. This section also describes how tire forces are computed for each program model.

Collision Forces provides an analysis of the collision model used by EDSMAC.

Numerical Integration provides an overview of the numerical integration routines used by EDSVS, EDVTS and EDSMAC.

Before addressing these technical issues, it is insightful to review the historical development of simulations.

History of Simulations

The development of simulation programs parallels the development of the computer. Although analog computers were first

developed during the early 1900s, the first practical computers saw use during World War II. The aerospace industry was the first commercial user of digital computers in the 1950s [1]. These computers were used to used to simulate an aircraft before building it, thus reducing development costs.

In the late 1950s, the US Weather Bureau began developing weather simulation programs for weather forecasting [2]. These programs were remarkably similar to today's simulation programs: They used initial conditions (wind velocities and pressures) supplied by weather ballons and fluid dynamics to simulate the flow in the upper atmosphere. The integration time step was several minutes!

Use in Accident Investigation

The literature suggests the first to develop motor vehicle simulations useful in the study of highway safety was McHenry in 1968 [3] at Cornell Aeronautical Lab. This research evolved into the Highway-Vehicle-Object Simulation Model (HVOSM), a sophisticated 3-dimensional computer simulation, in 1976 [4]. During the same era, McHenry also produced the Simulation Model of Automobile Collisions (SMAC), a 2-dimensional motor vehicle crash simulation [5]. McHenry and others at Cornell Aeronautical Lab were also developing crash victim simulation programs, the first of which was produced in 1966 [6]. The primary funding for these programs came from the National Highway Traffic Safety Administration, a federal agency within the U.S. Department of Transportation.

A parallel effort at the University of Michigan produced several commercial vehicle simulations. The first was by Moncarz et al [7], followed by more sophisticated simulations by MacAdam et al [8]. Michigan also produced occupant simulations [9]. The primary funding for these programs came from the Motor Vehicle Manufacturer's Association.

All the above simulations were developed for use on IBM Model 360 (or similar) mainframe computers. When the IBM Personal Computer was introduced in 1981, it became possible to modify

these programs and make them available for use by private accident investigators.

EDSVS and EDVTS [10,11] were produced from Moncarz' TBST and TBSTT programs, respectively. EDSMAC [12] was produced from the SMAC program. In each case, the program was developed by stripping away the mainframe-oriented input and output routines, while saving the main calculation portion of the program. The input and output routines were then replaced with routines which take advantage of the personal computer's friendly environment. Validation studies were performed to insure the PC versions produced the same results as their mainframe counterparts.

Scope of this Publication

This publication describes the calculations performed during the main calculation phase of the EDSVS, EDVTS and EDSMAC computer simulation programs. The theoretical developments are presented for a thorough understanding of the analysis. For details on the user interface, the reader is referred to their respective Program Manuals [10,11,12]. For details on how the theoretical models have been programmed, the interested reader is referred to the specific source code listings [13,14,15].

ANATOMY OF A SIMULATION

In this chapter, we will discuss the design of simulation programs. We will begin by describing in general terms the features common to all simulation programs. Then, we will describe the specific design of EDSVS, EDVTS and EDSMAC.

Simulation programs are used as mathematical models to predict the outcome of a dynamic event. Examples of such events include weather forecasting, traffic flow predictions, aerodynamic modeling of experimental aircraft and the dynamic modeling of motor vehicle accidents. Simulations use a mathematical or physical model to estimate the conditions at a discrete point in time; then, a numerical integration routine is used to calculate the first and second integrals relating to the physical conditions. In the case of accident simulations, these physical conditions are the accelerations experienced at the given point in time; the first and second integrals are the velocities and positions, respectively. The numerical integration routine then increments the simulation time by a discrete, user-specified time step and the calculations are repeated. Obviously, simulation programs perform repetitive, math-intensive calculations which can result in significant computation time.

Program Elements

All simulation programs have a similar structure which includes the following program elements:

- Controlling Program
- Numerical Integration
- Force Computations
- Derivative Computations

The relationship between these elements is shown in figure 1. The purpose of each of these elements is described below.

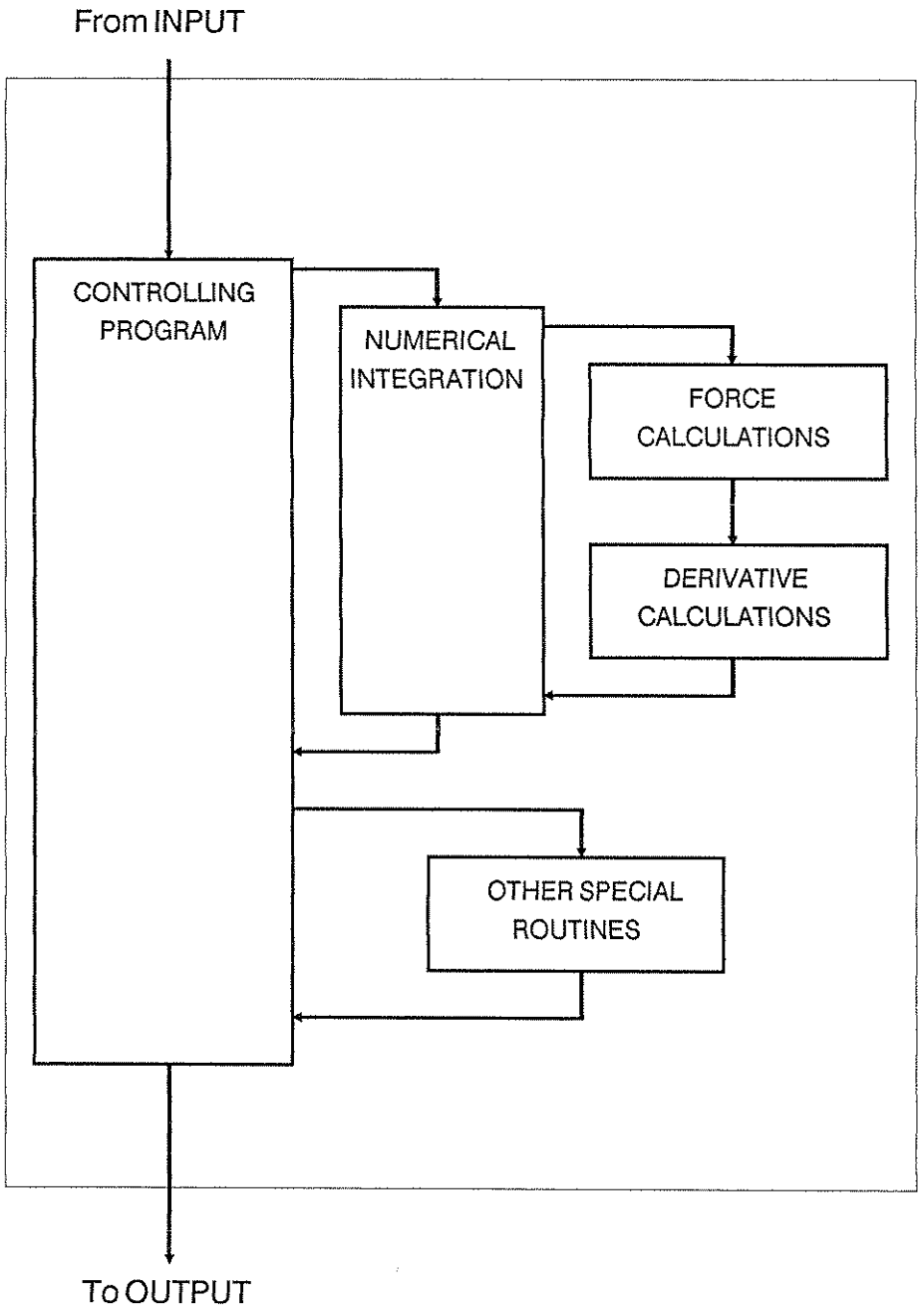

Figure 1 - Simulation Program Elements

Controlling Program

The controlling program does just what its name suggests: It controls the entire sequence of calculations. The following tasks are typically performed by the controlling program:

- initializes all variables
- calls the numerical integration routine
- calls the output routine
- terminates execution

For programs such as EDSVS and EDVTS, most of this control is actually performed by the numerical integration program; thus the control program is quite small. On the other hand, EDSMAC has a sophisticated and rather complicated control program which, in addition to the above events, must also determine when vehicle interaction occurs and which accident phase (pre-impact, impact, or post-impact) is currently being simulated.

Numerical Integration

The numerical integration routine is used to integrate the equations of motion. It starts with the accelerations supplied by the derivative calculations routine (below). The simulation time is increased by some small fraction of the user-specified time step. The velocities (first integral of the acceleration during the time increment) and positions (second integral of the acceleration during the time increment) are then computed.

The numerical integration routine then returns to the control program and the entire sequence of calculations is repeated.

Force Computations

The force computation routine performs a free body analysis of the vehicle at the current time step. Therefore, a physical/mathematical model is included in this routine. This model must be

general enough to describe the forces acting on the object under all anticipated conditions.

The complexity of the physical model is determined by its number of *degrees of freedom*. In general, the number of degrees of freedom is determined by the number of masses being simulated and the allowable motion for each mass. For example, a vehicle modeled as a single mass on a 2-dimensional flat surface has 3 degrees of freedom. These are linear motion in the earth-fixed X and Y directions and rotation about the Z axis (see figure 2). If the vehicle model is extended to 3 dimensions, the number of degrees of freedom increases to 6. The additional degrees of freedom include linear motion of the vehicle mass in the Z direction and rotation about the X and Y axes, as shown in figure 2. If suspension motion (jounce/rebound) and wheel spins are included, the number of degrees of freedom increases to 14 (vertical and angular motion at each of four wheels; see figure 2).

The number of degrees of freedom also determines the number of input parameters required to describe the vehicular motion. Most simulation programs are a compromise between the number of degrees of freedom required for satisfactory accuracy and the number of available input variables.

EDSVS and EDSMAC analyze 3 degrees of freedom per vehicle. EDVTS analyzes an additional degree of freedom, the articulation angle of the trailer.

The final output of the force computations routine is the summation of external forces and moments acting on each mass.

Derivative Calculations

The derivative calculation routine uses the summation of forces and moments produced in the Force Caclulation routine and applies Newton's 2nd Law ($\Sigma F = ma$ and $\Sigma M = I\alpha$) to calculate the resulting linear and angular accelerations.

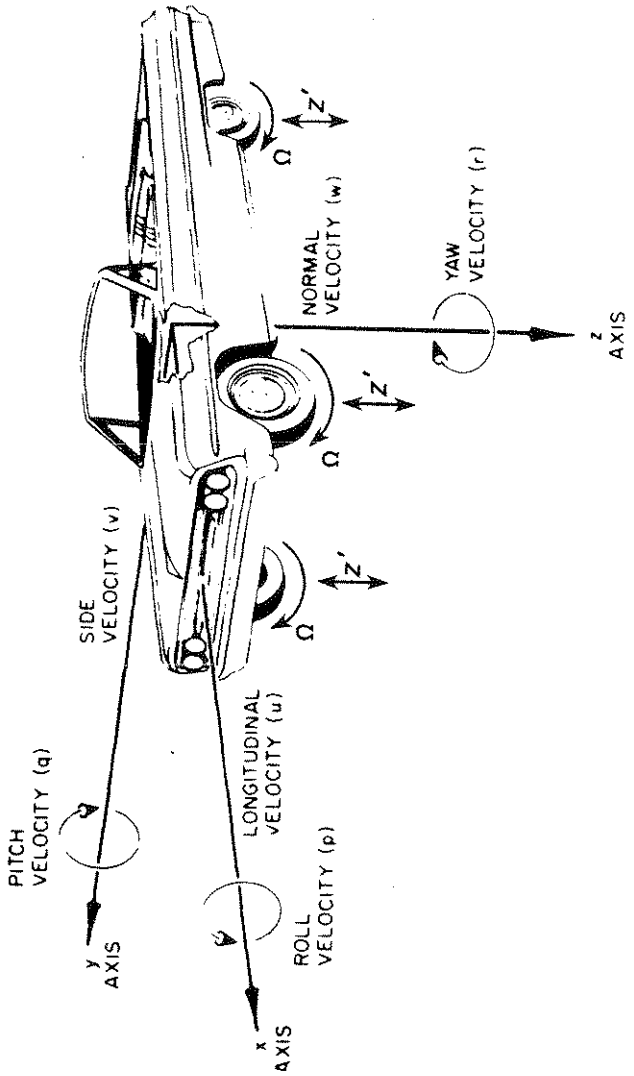

Figure 2 - Degrees of Freedom

SPECIFIC FLOW CHARTS

Program flow charts for EDSVS, EDVTS and EDSMAC are described below.

EDSVS Flow Chart

The EDSVS flow chart is shown on the following page. It contains six major routines:

- INITIALIZE
- HPCG
- FCT
- TIRE
- WRITE.OUTPUT
- SHUT.DOWN

The *INITIALIZE* routine loads the input data file and initializes all variables, converting from user units to program units (i.e., inches to feet and degrees to radians). The initial wheel loads and combined (vehicle plus payload) weight, yaw moment of inertia and CG height are also computed here.

The *HPCG* routine is the numerical integration routine used by EDSVS. In addition to its normal duties of calling the force-computing algorithm (in this case, FCT) and updating the time step, it also determines when an output record is written, a job usually accomplished by the controlling program.

The flow chart reveals a feature contained in HPCG not contained in most numerical integration routines: it monitors the difference between the predicted derivative vector and the actual derivative vector (hence it is called a 'predictor-corrector' method).

The *FCT* routine contains the equations of motion for the 3 degree-of-freedom (forward, lateral and angular plane motion) model used by EDSVS. It includes the driver input tables to deter-

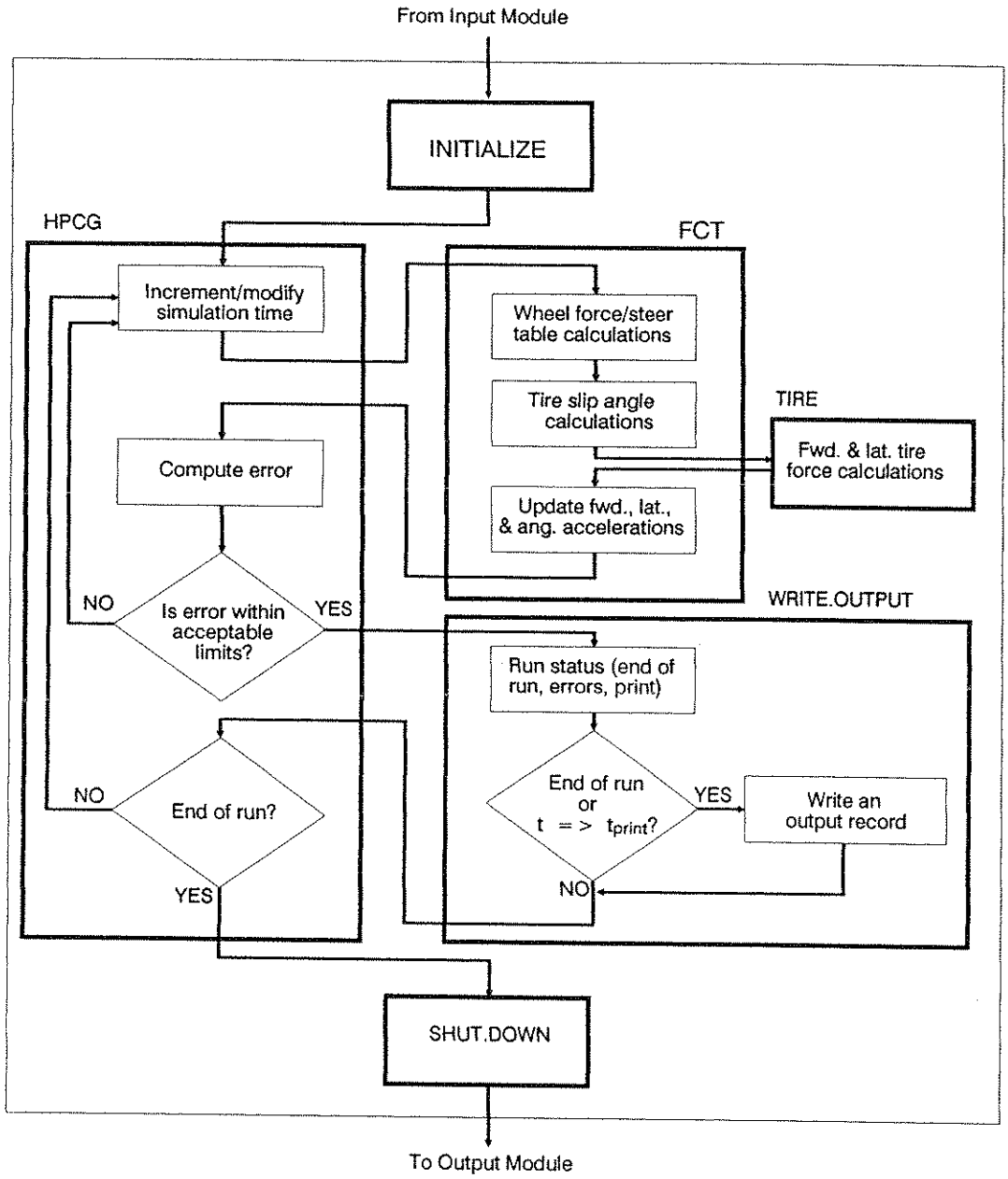

Figure 3 - EDSVS Flow Chart

mine the current level of accelerating, braking and/or steering at each wheel. It uses analytic geometry to determine the current slip angles at each wheel. It uses the weight dictribution to compute the current tire loads. Then it calls the TIRE routine (see below). Finally, with the new tire forces and vehicle inertia (total vehicle mass and yaw moment), it computes the new levels of linear and angular acceleration.

The *TIRE* routine is called by FCT (see above). It uses the coefficients of friction, tire cornering stiffnesses and current vertical tire loads and slip angles to compute the forward and lateral forces at each tire.

The *WRITE.OUTPUT* routine checks the status of the run (i.e., has the vehicle stopped? Has the user pressed ESCape? Has a wheel lifted off the ground?) and sets appropriate flags. It then writes an output record containing the position, velocity and acceleration variables, as well as other variables, on the output file.

The *SHUT.DOWN* routine writes any warning messages to the output file and closes it. Control is then transfered to the Output Session where the user views the results.

EDVTS Flow Chart

The EDVTS flow chart is nearly identical to the EDSVS flow chart. As shown on the following page, it contains the same six major routines. However, because EDVTS has an additional degree of freedom, (the articulation angle of the trailer), an additional seventh routine, called SIMSOL, is used for upating the linear and angular accelerations.

EDSMAC Flow Chart

The EDSMAC program calculations occur in two major steps:

- Main Simulation Loop
- Vehicle Damage Summary

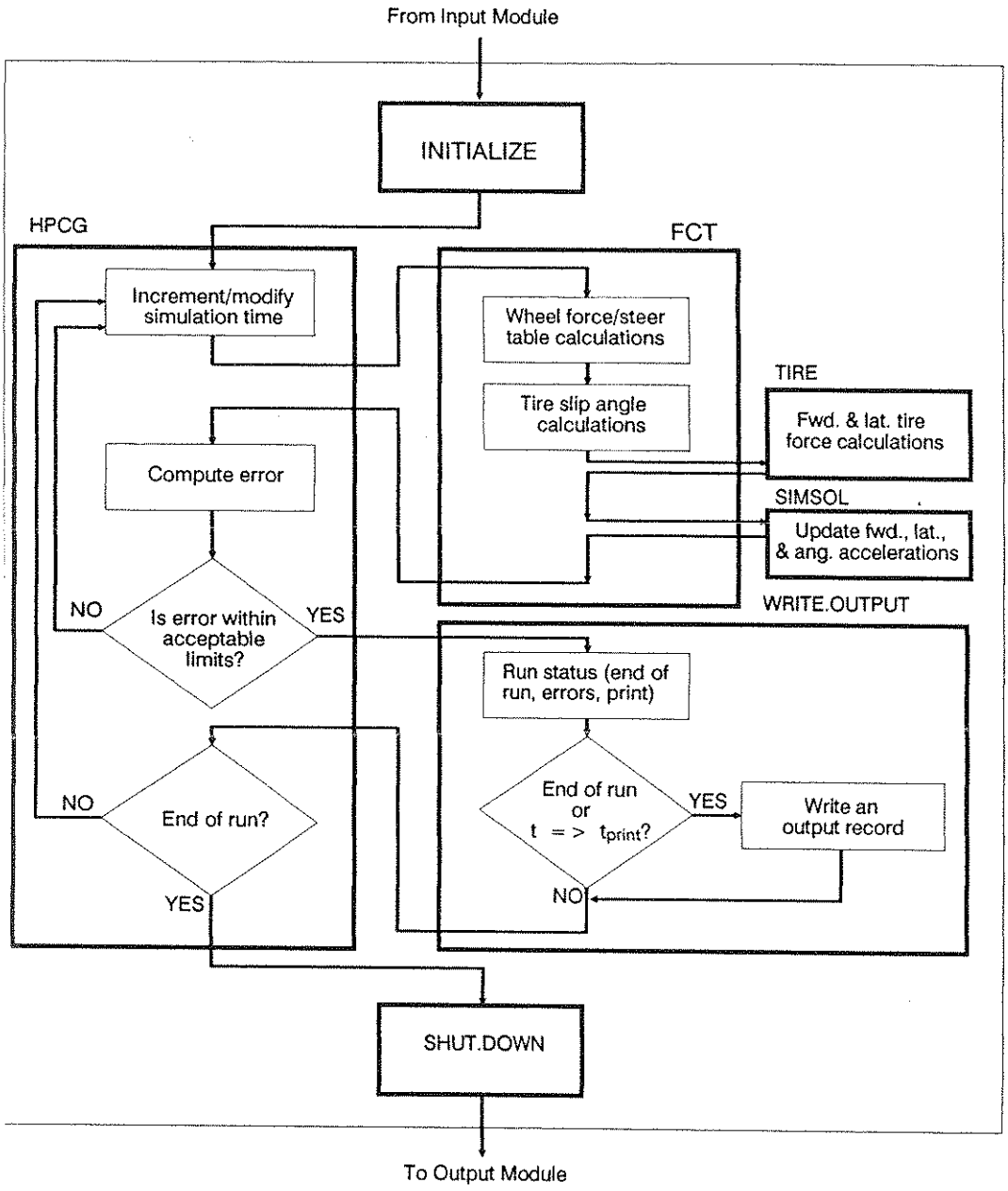

Figure 4 - EDVTS Flow Chart

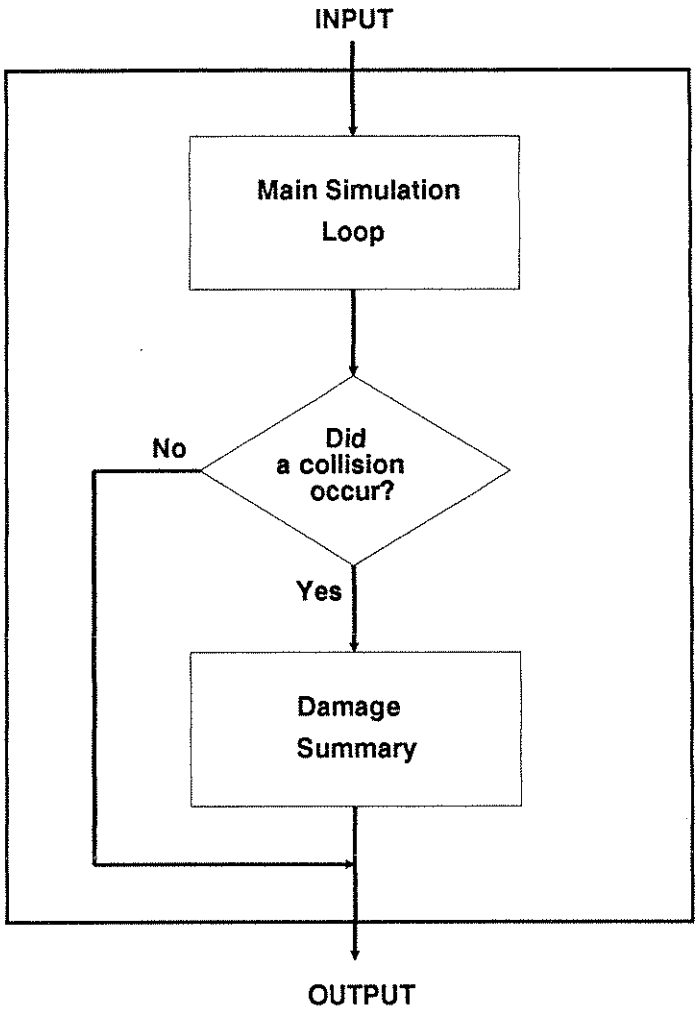

Figure 5 - EDSMAC Flow Chart

An overall calculation phase flow chart is shown on above. Because each step is rather complex, and because they occur separately, each step will be described separately.

Main Simulation Loop

Most of the calculations are performed here. Inspection of the Main Simulation Loop flow chart (see figure 6) reveals there are

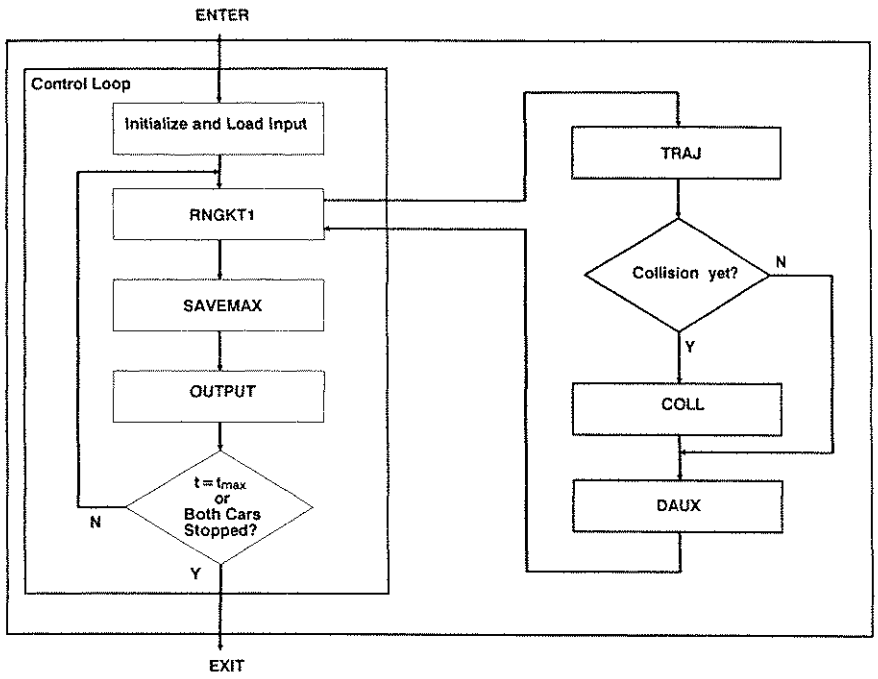

Figure 6 - EDSMAC Main Simulation Loop Flow Chart

actually four major procedures within this loop. These procedures are:

- Control Loop
- TRAJ
- COLL
- DAUX

The control loop loads the input data file and initializes all variables. It then calls RNGKT1, the numerical integration procedure, which returns with the updated vehicle positions and velocities based on the current accelerations.

The control loop calls a routine named SAVEMAX, which compares the current accelerations with previous accelerations and stores the magnitude and direction of any new maxima which have occurred.

When the output print time is reached, the control loop writes an output record containing the vehicle positions, velocities and accelerations for the current simulation time. Wheel positions and skid flags are also written. These are used during the Graphics Session to draw skidmarks.

The *TRAJ* (trajectory) routine computes the forward and lateral tire forces at each wheel for the current timestep. The individual forces are then resolved as a single force and moment acting at the vehicle CGs.

The *COLL* (collision) routine computes the forward and lateral collision forces at the CG of each vehicle for the current timestep.

The *DAUX* routine calculates the total force and moment (the sum of the tire and collision forces and moments) acting on the vehicle for the current timestep. It then applies Newton's 2^{nd} law to determine the total linear and angular accelerations for each vehicle.

The Main Simulation Loop continues until the vehicles come to rest or the maximum simulation time is reached.

Vehicle Damage Summary

The Vehicle Damage Summary is a routine called *SMACDAM*. It is called at the conclusion of the Main Simulation Loop. The Vehicle Damage Summary flow chart (see figure 7) reveals it contains only two major routines:

- RANGDAM
- DAMAGE

The *RANGDAM* routine analyzes and organizes the table of collision vectors computed during the Main Simulation Loop. The final result is a series of individual damage ranges, each having its own Collision Deformation Classification (CDC) and delta-V.

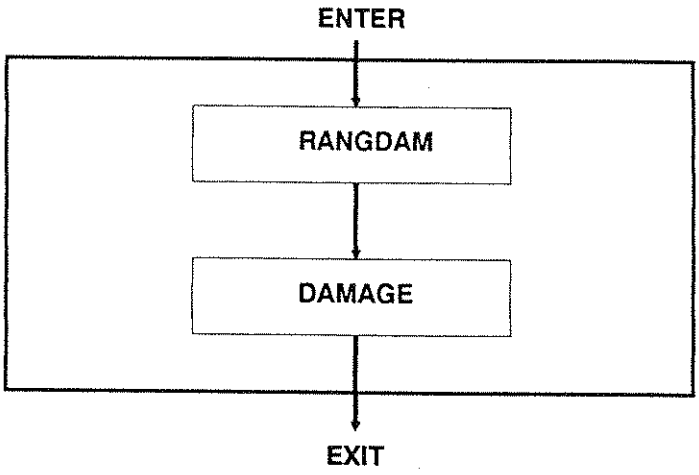

Figure 7 - EDSMAC Vehicle Damage Summary Flow Chart

The *DAMAGE* routine analyzes the collision vectors to assign a CDC to each damage range.

For further information about the major calculation procedures, refer to the specific chapters regarding tire and collision force calculations.

VEHICLE MODELS

Perhaps the most important part of a simulation program is the physical model itself. The physical model is a general, free-body analysis of the simulated object. This analysis must define the locations of all external forces (typically tire and collision forces) to be considered. It then accesses the routines necessary to compute these forces. Finally, the vehicle model computes the resultant moments and forces acting at the object's center of gravity.

This chapter describes the models used by the EDSVS, EDVTS and EDSMAC programs.

EDSVS

EDSVS is a 3 degree-of-freedom model which simulates the motion of a unit (single mass) vehicle in the X-Y plane. Thus, the vehicle moves only in the X and Y directions and rotates only about the Z (yaw) axis. Any motion in the X-Z and Y-Z planes, or rotation about the x (roll) or y (pitch) axes is ignored.

The routine in EDSVS which computes the exterior forces applied to the vehicle is called FCT. This routine involves the following procedures each time it is called:

- compute the slip angle at each tire
- compute the vertical wheel loads from the static load and the current longitudinal and lateral load transfers
- call the tire routine
- sum the external moments and forces
- compute the resulting acceleration

Each of these tasks is described below.

Computing Slip Angles

The relationship used for computing the slip angle at each tire is shown in figure 8. This relationship is based on vehicle geometry and kinematics, as well as the front axle steer angle, δ (which is obtained from the user-entered steer table). Slip angles for left side and right side tires are assumed to be equal - a good assumption as long as the track width is small compared to the turn radius. The slip angle for each tire is computed as follows:

Front tires -

$$\alpha_{1,2} = \text{ATAN2}((v + a\dot{\psi})/u) - \delta$$

Rear or front tandem axle tires -

$$\alpha_{3,4} = \text{ATAN2}((v - (b - (At/2))\dot{\psi})/u)$$

Trailing tandem axle tires (skip if tandems not present) -

$$\alpha_{5,6} = \text{ATAN2}((v - (b + (At/2))\dot{\psi})/u)$$

NOTE: The ATAN2 function is a special arctangent function having the entire unit circle as its range and domain. Thus, it is capable of returning correct results for angles in all four quadrants.

Computing Vertical Wheel Loads

The vertical wheel loads are computed as the sum of the static load and the longitudinal and lateral load transfers at each wheel. The static load is computed once at the beginning of execution. The load transfers are computed at each time step. This section describes the calculation of these forces.

Static Load

The static load is simply a function of the weight distribution and the lateral offset of any payload. Taking the sum of moments and forces in the Z direction, the static load at each wheel is computed.

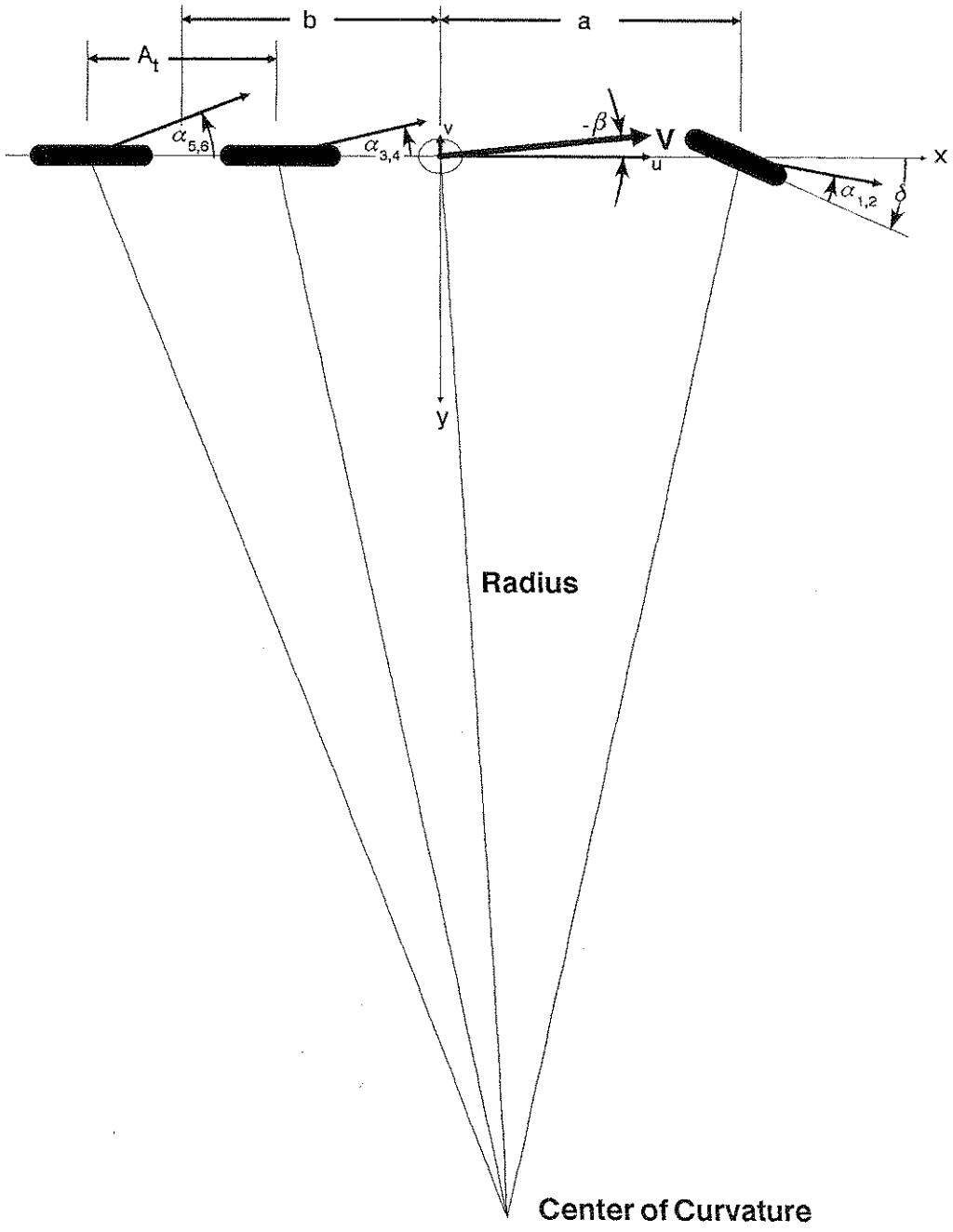

Figure 8 - Kinematic Relationship for Computing Slip Angles

The free-body diagram shown in figure 9 illustrates the vehicle configuration and the applied forces.

To calculate the static wheel loads, first calculate the lateral weight shift at the front and rear axles due to the lateral payload offset, Δy,

$$\Delta W_{y,front} = \gamma_1 \Delta y W / T_1$$

$$\Delta W_{y,rear} = (1-\gamma_1) \Delta y W / T_2$$

The vertical load at each wheel is then computed as follows

$$F1_{z,static} = bW/(2(a + b)) - \Delta W_{y,front}$$

$$F2_{z,static} = F1_{z,static} + 2\Delta W_{y,front}$$

$$F3_{z,static} = aW/(2(a + b)) - \Delta W_{y,rear}$$

$$F4_{z,static} = F3_{z,static} + 2\Delta W_{y,rear}$$

If the rear has tandem axles, the loads at wheels 3 and 5 are split equally, as are the loads at wheels 4 and 6.

Lateral Load Transfer

Lateral load transfer at the front axle is the fraction, γ_1, of the total lateral load transfer:

$$\Delta F_z = (\gamma_1 ZM/T_f) A_{lat},$$

where

- γ_1 = lateral load transfer coefficient (fraction of lateral load transferred to the front axle)
- Z = elevation of CG
- T_f = front track width
- $A_{lat} = \dot{v} + u\dot{\psi}$

Thus, for the front axle, the vertical load on the outside and inside tires will be increased and decreased, respectively by the amount ΔF_z.

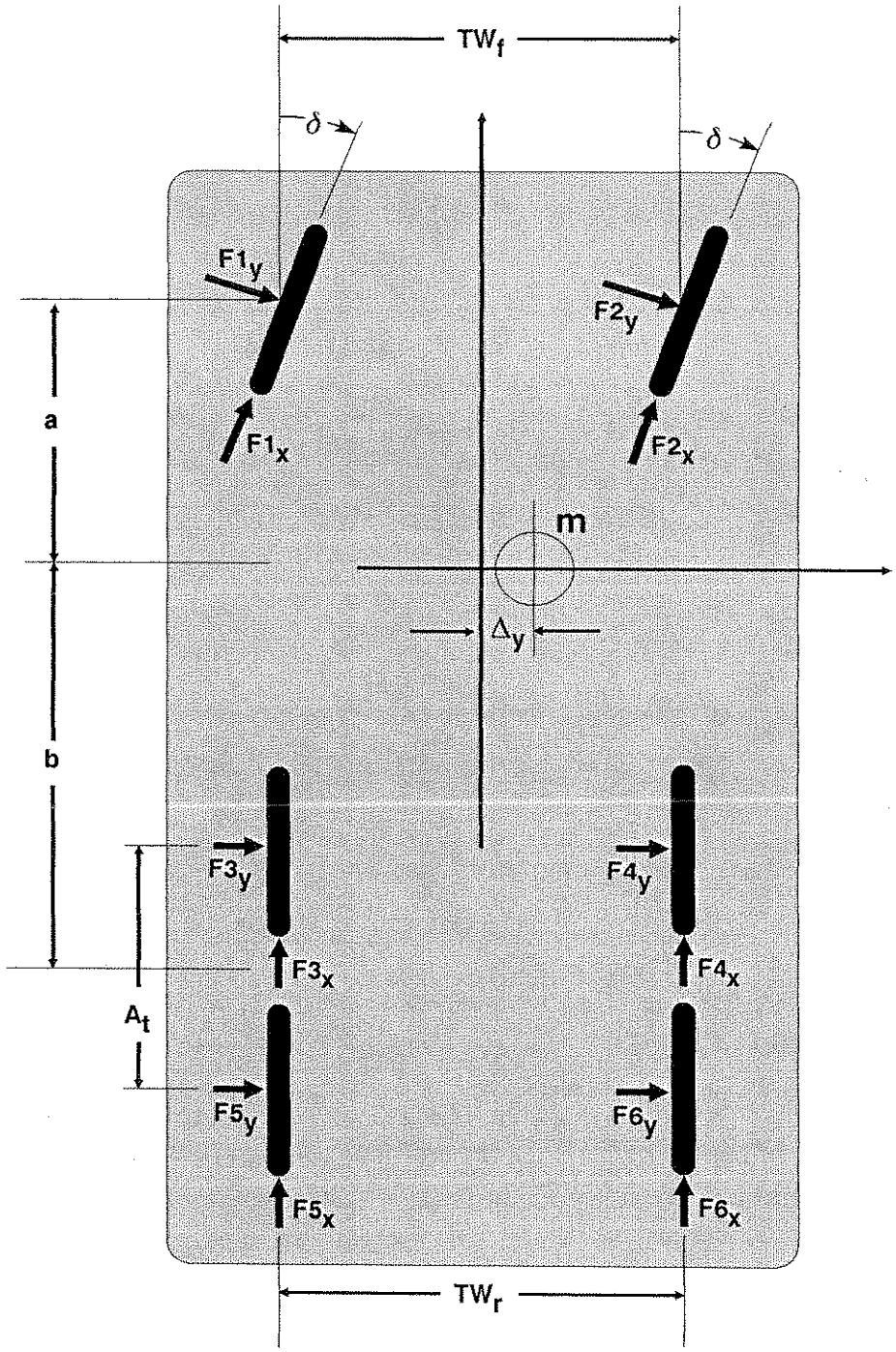

Figure 9 - Free-body Diagram of EDSVS Vehicle Model

The lateral load transfer on the rear tires is

$$\Delta F_z = ((1-\gamma_1)ZM/T_f)A_{lat},$$

Again, the vertical tire load on the outside tires increases and the load on the inside tires descreases by ΔF_z.

Longitudinal Load Transfers

Longitudinal load transfers are computed by summing the moments and forces in the vehicle-fixed x direction. The longitudinal load transfer to the front axle is

$$\Delta F_{z,front} = -(1/2(a+b))(ZmA_{long} + \gamma_3(F3_x + F4_x + F5_x + F6_x)A_t)$$

where

$$A_{long} = \dot{u} - v\dot{\psi}$$

The longitudinal load transfer to the rear axle is equal to the load added to or removed from the front. Thus,

$$\Delta F_{z,rear} = -\Delta F_{z,front}$$

If the vehicle has tandem rear axles, the rear axle load transfers are modified, first by splitting the vertical load between the axles then by accounting for the inter-axle load transfer as follows:

$$\Delta F_{z,front\ tandem} = \Delta F_{z,rear}/2 - \gamma_3(F3_x + F4_x + F5_x + F6_x)$$
$$\Delta F_{z,rear\ tandem} = \Delta F_{z,rear}/2 + \gamma_3(F3_x + F4_x + F5_x + F6_x)$$

The total vertical load on each wheel is the sum of the static load and the longitudinal and lateral load transfers,

$$F_z = F_{z,static} + \Delta F_{z,lateral} + \Delta F_{z,longitudinal}$$

These vertical tire forces are passed to the TIRE routine (see **Tire Forces**), which calculates the forward and lateral forces, F_x and F_y at each tire. The last step is to sum these forces and moments to compute the resulting vehicle motion.

Sum of the External Forces and Moments

All external forces act at the tire-road shear interface. These forces have been computed by the TIRE routine. Summing these forces in the vehicle-fixed x and y directions and about the z axis, we have

$$\Sigma F_X = F1_X\cos\delta + F2_X\cos\delta - F1_Y\sin\delta - F2_Y\sin\delta + F3_X + F4_X + F5_X + F6_X$$

$$\Sigma F_Y = F1_X\sin\delta + F2_X\sin\delta + F1_Y\cos\delta + F2_Y\cos\delta + F3_Y + F4_Y + F5_Y + F6_Y$$

and

$$\begin{aligned}\Sigma M_Z = {} & a[(F1_X + F2_X)\sin\delta + (F1_Y + F2_Y)\cos\delta] \\ & + [(T_f/2)(F1_X - F2_X) + \Delta_Y(F1_X + F2_X)]\cos\delta \\ & - [(T_f/2)(F1_Y - F2_Y) + \Delta_Y(F1_Y + F2_Y)]\sin\delta \\ & - [(b - A_t/2)(F3_Y + F4_Y) + (b + A_t/2)(F5_Y + F6_Y)] \\ & + T_r/2(F3_X + F5_X - F4_X - F6_X) \\ & + \Delta_Y(F3_X + F5_X + F4_X + F6_X)\end{aligned}$$

Computing the Acceleration

The acceleration of the vehicle is computed from Newton's 2nd law. These accelerations are derived with respect to the earth-fixed coordinate system. However, all the forces are with respect to the vehicle-fixed coordinate system. Thus, a transformation is required.

In the earth-fixed X direction, $\Sigma F_X = ma_X = m(\dot{u} - v\dot{\psi})$. Thus by rearranging, we obtain the acceleration in the X direction,

$$\dot{u} = \Sigma F_X/m + v\dot{\psi}$$

Similarly, in the Y direction we have $\Sigma F_Y = ma_Y = m(\dot{v} + u\dot{\psi})$. Thus,

$$\dot{v} = \Sigma F_Y/m - u\dot{\psi}$$

And about the Z axis, $\Sigma M_Z = I\alpha$ (note that the angular acceleration is identical in both the earth-fixed and vehicle-fixed coordinate systems). Solving for angular accelerations yields

$$\alpha = \Sigma M_Z/I$$

These linear and angular accelerations are passed to the numerical integration routine for further processing to determine the velocities and positions at the next time step.

EDVTS

EDVTS is a four degree-of-freedom model which models the motion of a tow vehicle and trailer in the X-Y plane. The combination may be a passenger car towing a small trailer or a tractor pulling a semi-trailer; the euqations of motion are the same. The tow vehicle moves only in the X and Y directions and rotates only about the Z (yaw) axis). The trailer is free only to articulate about its connection to the tow vehicle. Any motion by either mass in the X-Z or Y-Z plane, or rotation about the vehicles' x (roll) or y (pitch) axes is ignored.

The routine in EDVTS which computes the exterior forces applied to the vehicle is called FCT. The resulting accelerations are computed here also. This routine is very similar to EDSVS's FCT routine except that an additional procedure is used to handle the accelerations. The FCT routine used by EDVTS involves the following procedures each time it is called:

- compute the sideslip angles at each tire
- compute the vertical wheel loads from the static load and the current longitudinal and lateral load transfers
- call the TIRE routine
- sum the external moments and forces
- compute the resulting acceleration

Each of these tasks is described below.

Computing the Slip Angles

The relationship used for computing the slip angle at each tire is shown in figure 10. This relationship is based on vehicle geometry and kinematics, as well as the front axle steer angle, δ (which is obtained from the user-entered steer table). Slip angles for left side and right side tires are assumed to be equal - a good assumption as

long as the track width is small compared to the turn radius. The slip angle for each tire is computed as follows:

Tow vehicle front tires -

$$\alpha_{1,2} = \text{ATAN2}((v + a\dot{\psi})/u) - \delta$$

Tow vehicle rear or front tandem axle tires -

$$\alpha_{3,4} = \text{ATAN2}((v - (b - (A_t/2))\dot{\psi})/u)$$

Tow vehicle trailing tandem axle tires (skip if tandems not present) -

$$\alpha_{5,6} = \text{ATAN2}((v - (b + (A_t/2))\dot{\psi})/u)$$

Trailer front tandem axle tires -

$$\alpha_{7,8} = \text{ATAN2}(((v - h\dot{\psi})\cos\gamma - (c + (d - A_{tt}/2))(\dot{\psi} + \dot{\gamma}) - v\sin\gamma)/(u\cos\gamma + v - h\dot{\psi}\sin\gamma)$$

Trailer trailing tandem axle tires (skip if tandems not present) -

$$\alpha_{9,10} = \text{ATAN2}(((v - h\dot{\psi})\cos\gamma - (c + (d + A_{tt}/2))(\dot{\psi} + \dot{\gamma}) - v\sin\gamma)/(u\cos\gamma + v - h\dot{\psi}\sin\gamma)$$

Computing Vertical Wheel Loads

The vertical wheel loads are computed as the sum of the static load and the longitudinal and lateral load transfers at each wheel. The static load is computed once at the beginning of execution. The load transfers are computed at each time step. This section describes the calculation of these forces.

Static Load

The static load is simply a function of the weight distribution and the lateral offset of any payload. Taking the sum of moments and forces in the Z direction, the static load at each wheel is computed.

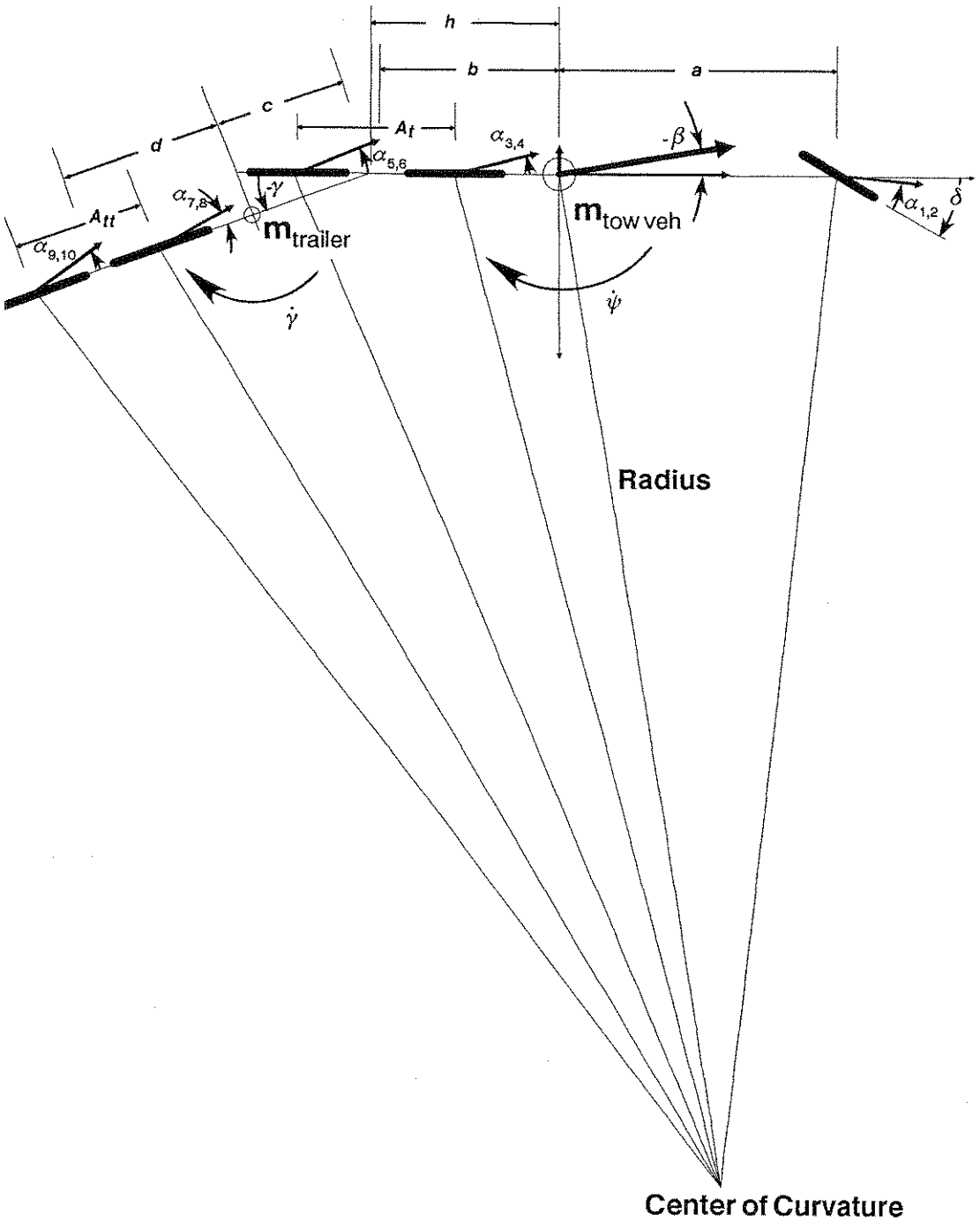

Figure 10 - Computing Slip Angles for Vehicle-Trailer

The free-body diagram shown in figure 11 illustrates the vehicle configuration and the applied forces.

The vertical load at each wheel of the tow vehicle is computed as follows:

$$F1_{z,static} = bW_1/(2(a + b)) + dW_2(b-h)/2(a+b)(c+d)$$

$$F2_{z,static} = F1_{z,static}$$

$$F3_{z,static} = aW_1/(2(a + b)) + dW_2(a+h)/2(a+b)(c+d)$$

$$F4_{z,static} = F3_{z,static}$$

If the rear has tandem axles, the loads at wheels 3 and 5 are split equally, as are the loads at wheels 4 and 6.

To calculate the static wheel loads on the trailer, first calculate the lateral weight shift at due to payload offset, Δy,

$$\Delta W_y = \Delta y W_2/T_3$$

The vertical load at each wheel is then computed as follows:

$$F7_{z,static} = cW_2/(2(c + d)) - \Delta W_y$$

$$F8_{z,static} = F7_{z,static} + 2\Delta W_y$$

If the trailer has tandem axles, the loads at wheels 7 and 9 are split equally, as are the loads at wheels 8 and 10.

Inertial Mass Forces

Before computing longitudinal and lateral load transfers, the hitch force in the vehicle-fixed x and y directions must be computed. To simplify calculations, these forces are added together with the inertial forces. The tow vehicle inertial forces in the earth-fixed X and Y directions are:

$$\lambda_1 = -m_2(A_{long} + h\dot{\psi}^2)\cos\gamma - m_2(A_{lat} - h\ddot{\psi})\sin\gamma - \overline{C}m_2(\dot{\psi} + \dot{\gamma})^2$$

$$\lambda_2 = m_2(A_{long} + h\dot{\psi}^2)\sin\gamma - m_2(A_{lat} - h\ddot{\psi})\cos\gamma + \overline{C}m_2(\dot{\psi} + \dot{\gamma})$$

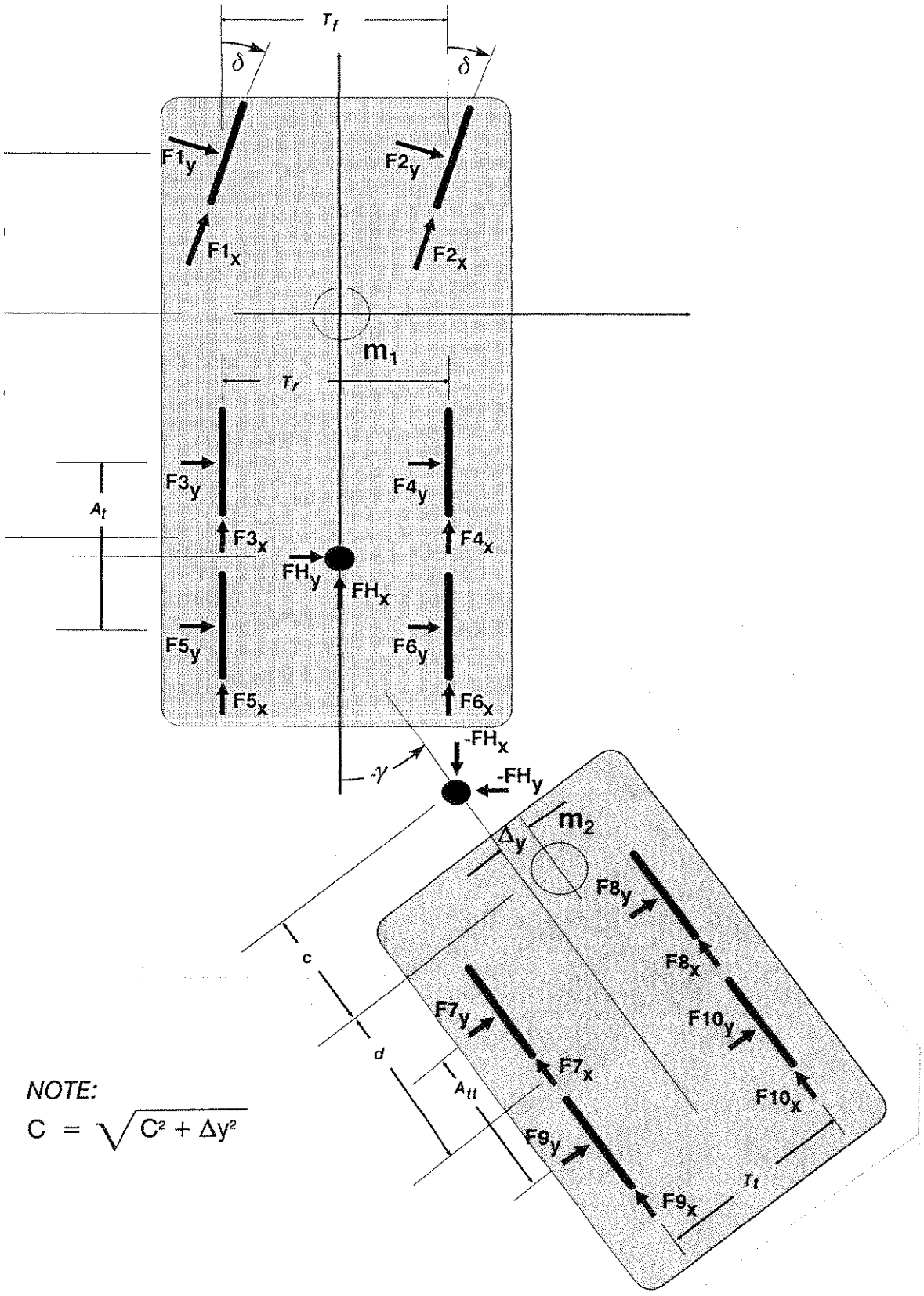

Figure 11 - Free-body Diagram of EDVTS Vehicle Model

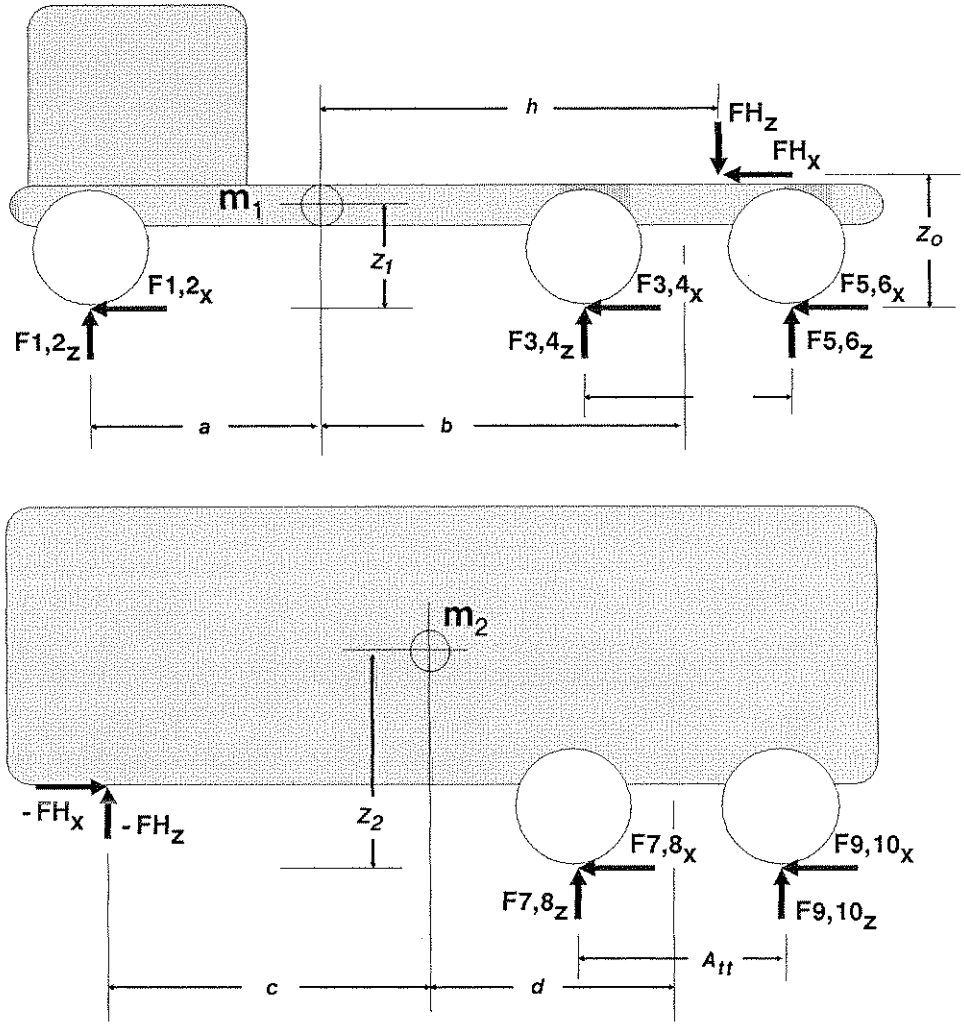

Figure 11 - EDVTS Free-body Diagram (cont)

The trailer inertial forces in the earth-fixed X and Y directions are:

$$\lambda_3 = -m_2(A_{long} + h\dot{\psi}^2 - \overline{C}m_2(\dot{\psi} + \dot{\gamma}))\sin\gamma - \overline{C}m_2(\dot{\psi} + \dot{\gamma})^2\cos\gamma$$

$$\lambda_4 = -m_2(A_{lat} - h\dot{\psi}) + \overline{C}m_2(\dot{\psi} + \dot{\gamma})\cos\gamma - \overline{C}m_2(\dot{\psi} + \dot{\gamma})^2\sin\gamma$$

Forces and Moments at the King Pin or Ball

Forces applied to the tow vehicle at the trailer connection are computed from the trailer tire forces and inertial forces. These longitudinal and lateral connection forces are:

$$FH_x = \lambda_3 + F7_x + F8_x + F9_x + F10_x$$

$$FH_y = \lambda_4 + F7_y + F8_y + F9_y + F10_y$$

The torque, T, acting about the connection axis is

$$T = 2\mu_c d W_2 R_c/3(c+d) \qquad \text{for } \dot{\gamma} > 1.72 \text{ deg/sec}$$

$$= \dot{\gamma}/33.3(2\mu_c d W_2 R_c/3(c+d)) \qquad \text{for } \dot{\gamma} \leq 1.72 \text{ deg/sec}$$

where

μ_c = friction at connection
R_c = radius of connection

Lateral Load Transfer

Now all the exterior forces are known and the load transfers can be computed. The load transfer at the front axle of the tow vehicle is the fraction, γ_1, of the total lateral load transfer:

$$\Delta F_z = (\gamma_1 Z_1 m_1/T_f)A_{lat} - \gamma_1 Z_0 FH_y/T_f$$

where

γ_1 = lateral load transfer coefficient (fraction of lateral load transfered to the front axle)
Z_0 = elevation of trailer connection
Z_1 = elevation of tow vehicle CG
FH_y = lateral component of force at trailer connection
T_f = tow vehicle front track width
A_{lat} = $\dot{v} + u\dot{\psi}$

Thus, for the front axle, the vertical load on the outside and inside tires will be increased and decreased, respectively by the amount ΔFz.

The lateral load transfer on the rear tires is

$$\Delta F_z = ((1-\gamma_1)Z_1 m_1/T_r)A_{lat} - (1-\gamma_1)Z_0 FH_y/T_r$$

where

T_r = tow vehicle rear track width

Again, the vertical tire load on the outside tires increases and the load on the inside tires decreases by ΔF_z. If the tow vehicle has tandem rear axles, the load transfer is split between the leading and trailing axles.

The lateral load transfers at the trailer tires are:

$$\Delta F_z = -(Z_2-Z_0)(1-\lambda_1)/T_t + Z_0(F7y + F8y + F9y + F10y)/T_t$$

where

T_t = trailer track width

If the trailer has tandem axles, the lateral load transfer is split between the leading and trailing axles.

Longitudinal Load Transfers

Longitudinal load transfers are computed by summing the moments and forces in the vehicle-fixed X direction. First, the vertical load transfer at the trailer connection, ΔFH_z, is computed:

$$\Delta FH_z = [(Z_2\cos\xi - Z_0)\lambda_1 - (F7_x + F8_x + F9_x + F10_x)Z_0 + \lambda_2 Z_2 \sin\xi + (F7_x + F8_x + F9_x + F10_x)\gamma_4 AA_t]/(c + d)$$

where

ξ = $\arctan(\Delta y/c)$

γ_4 = user-entered inter-tandem load transfer coefficient for trailer

Now, the longitudinal load transfer to the front axle of the tow vehicle is

$$\Delta F_{z,front} = (1/2(a+b))(\Delta FH_z(b\text{-}H)Z_1 m_1 A_{long} + FH_x \gamma_3(F3_x + F4_x + F5_x + F6_x))A_{tt}$$

where

A_{long} = $\dot{u} - v\dot{\psi}$

γ_3 = user-entered inter-tandem load transfer coefficient for tow vehicle

The longitudinal load transfer to the rear axle is equal to the vertical load transfer from the trailer plus the load added to or removed from the front. Thus,

$$\Delta F_{z,rear} = \Delta FH_z - \Delta F_{z,front}$$

If the tow vehicle has tandem rear axles, the rear axle load transfers are modified, first by splitting the vertical load between the axles then by accounting for any inter-axle load transfer according to the user-entered longitudinal load transfer coefficient, γ_3:

$$\Delta F_{z,front\ tandem} = \Delta F_{z,rear}/2 - \gamma_3(F3_x + F4_x + F5_x + F6_x)$$
$$\Delta F_{z,rear\ tandem} = \Delta F_{z,rear}/2 + \gamma_3(F3_x + F4_x + F5_x + F6_x)$$

Finally, the longitudinal load transfers at the rear trailer axle are computed. Note this load transfer is generally the opposite of the vertical load transfered to the trailer connection,

$$\Delta F_{z,trailer} = -\Delta FH_z$$

If there is a payload offset (i.e., $\Delta y \neq 0$), there is an additional load transfer,

$$\Delta F_{z,trailer} = \Delta F_{z,trailer} + ((\lambda_2 \cos\xi - \lambda_1 \sin\xi)(Z_2 - Z_1) + (F7_y + F8_y + F9_y + F10_y)Z_0)/T_t$$

Just as for the tow vehicle, if the trailer has tandem axles, the longitudinal load transfer is distributed between them according to the user-entered longitudinal load transfer coefficient, γ_4:

$$\Delta F_{z,\text{trailer front}} = \Delta F_{z,\text{trailer}}/2 - \gamma_4(F7_x + F8_x + F9_x + F10_x)$$
$$\Delta F_{z,\text{trailer rear}} = \Delta F_{z,\text{trailer}}/2 + \gamma_4(F7_x + F8_x + F9_x + F10_x)$$

Total Vertical Load

The total vertical load on each wheel is the sum of the static load and the longitudinal and lateral load transfers,

$$F_z = F_{z,\text{static}} + \Delta F_{z,\text{lateral}} + \Delta F_{z,\text{longitudinal}}$$

These vertical forces are passed to the TIRE routine (see **Tire Forces**), which calculates the forward and lateral forces, F_x and F_y at each tire. The last step is to sum these forces and resulting moments to compute the resulting vehicle motion.

Sum of the External Forces and Moments

All external forces act at the tire-road shear interface. These forces have been calculated by the TIRE routine. As was mentioned previously, the vertical tire loads are calculated only once per time step. However, the tire forces are updated for each iteration through the numerical integration routine (see **Numerical Integration**). As a result the derivatives and associated inertial forces, $\lambda_{2\text{-}4}$, must also be updated (λ_1 is not required for the remaining calculations). The required forces are vehicle-fixed, not earth-fixed as calculated previously. Therefore, the updated inertial forces have the unnecessary terms stripped. The resulting inertial forces are

$$\lambda_2 = m_2((h\dot{\psi}^2 - v\dot{\psi})\sin(\gamma-\xi) - u\psi\cos(\gamma-\xi))$$
$$\lambda_3 = -m_2((h\dot{\psi}^2 - v\dot{\psi} + \overline{C}(\dot{\psi} + \dot{\gamma})^2\cos(\gamma-\xi))$$
$$\lambda_4 = -m_2((h\dot{\psi}^2 + u\dot{\psi} + \overline{C}(\dot{\psi} + \dot{\gamma})^2\sin(\gamma-\xi))$$

Matrix Solution of Equations of Motion

The equations of motion for the 4 degree-of-freedom system are solved by applying Newton's 2nd law. The equations are:

$$\Sigma F_X = (m_1 + m_2)\dot{u} + (\bar{C}m_2 \sin\xi)\ddot{\psi} + (\bar{C}m_2 \sin\xi)\ddot{\gamma}$$

$$\Sigma F_Y = (m_1 + m_2)\dot{v} - (m_2(h + \bar{C}\cos\xi))\ddot{\psi} - (\bar{C}m_2 \cos\xi)\ddot{\gamma}$$

$$\Sigma M_Z = -(hm_2)\dot{v} + (I_1 + m_2(h^2 + \bar{C}h\cos\xi))\ddot{\psi} + (\bar{C}hm_2 \cos\xi)\ddot{\gamma}$$

$$\Sigma M_C = (\bar{C}m_2 \sin\xi)\dot{u} - (\bar{C}m_2 \cos\xi)\dot{v} + (I_2 + \bar{C}m_2(\bar{C} + h\cos\xi))\ddot{\psi} + (I_2 + \bar{C}^2 m_2)\ddot{\gamma}$$

The above equations can be expressed in matrix form Ax = b. In this form, *x* is the derivative vector (earth-fixed accelerations), *b* is the summation of external forces and moments and *A* is a 4X4 matrix containing the inertial properties. Using this format,

$$[A] = \begin{bmatrix} m_1 + m_2 & 0 & \bar{C}m_2 \sin\xi & \bar{C}m_2 \sin\xi \\ 0 & m_1 + m_2 & -m_2(h + \bar{C}\cos\xi) & -\bar{C}m_2 \cos\xi \\ 0 & -hm_2 & I_1 + m_2(h^2 + \bar{C}h\cos\xi) & h\bar{C}m_2 \cos\xi \\ \bar{C}m_2 \sin\xi & -\bar{C}m_2 \cos\xi & I_2 + \bar{C}m_2(\bar{C} + h\cos\xi) & I_2 + \bar{C}^2 m_2 \end{bmatrix}$$

Similarly, the earth-fixed acceleration vector, x, is

$$x = \begin{bmatrix} \dot{U} \\ \dot{V} \\ \ddot{\psi} \\ \ddot{\gamma} \end{bmatrix}$$

(Tow vehicle, X-dir)

(Tow vehicle, Y-dir)

(Tow vehicle, Angular)

(Trailer, Angular)

The external forces are the sum of all tire forces plus hitch forces. These form the right-hand side of the vector equation, *b*, where

$$b = \begin{bmatrix} \Sigma F_x & \Sigma F_y & \Sigma M_z & \Sigma M_c \end{bmatrix}$$

From the free-body diagram (see figure 11),

$$\begin{aligned}\Sigma F_x = {} & m_1 v\dot{\psi} + \lambda_3 + (F1_x + F2_x)\cos\delta - (F1_y + F2_y)\sin\delta \\ & + F3_x + F4_x + F5_x + F6_x \\ & + (F7_x + F8_x + F9_x + F10_x)\cos\gamma \\ & - (F7_y + F8_y + F9_y + F10_y)\sin\gamma\end{aligned}$$

$$\begin{aligned}\Sigma F_y = {} & -m_1 u\dot{\psi} + \lambda_4 + (F1_x + F2_x)\sin\delta + (F1_y + F2_y)\cos\delta \\ & + F3_y + F4_y + F5_y + F6_y \\ & + (F7_x + F8_x + F9_x + F10_x)(\sin\gamma + \sin\xi\cos\gamma) \\ & + (F7_y + F8_y + F9_y + F10_y)\cos\gamma\end{aligned}$$

$$\begin{aligned}\Sigma M_z = {} & -h\lambda_4 + a((F1_x + F2_x)\sin\delta + (F1_y + F2_y)\cos\delta) \\ & + T_f/2(F1_x - F2_x)\cos\delta + T_r/2(F3_x + F5_x - F4_x - F6_x) \\ & -h(F7_x + F8_x + F9_x + F10_x)\sin\gamma \\ & -T_f/2(F1_y - F2_y)\sin\delta \\ & -(b - A_t/2)(F3_y + F4_y) + (b + A_t/2)(F5_y + F6_y) \\ & -h(F7_y + F8_y + F9_y + F10_y)\cos\gamma + T\end{aligned}$$

$$\begin{aligned}\Sigma M_c = {} & -c\lambda_2 + T_t/2(F7_x + F9_x - F8_x - F10_x) \\ & + (F7_x + F8_x + F9_x + F10_x)(\Delta y - c\sin\xi) \\ & -(c + d - A_{tt}/2)(F7_y + F8_y) \\ & -(c + d + A_{tt}/2)(F9_y + F10_y) - T\end{aligned}$$

Computing the Acceleration

The accelerations in the x vector are computed using simultaneous solutions (four vector equations and four unknown accelerations). This solution is performed in a subroutine called SIMSOL.

EDSMAC

EDSMAC is a 3 degree-of-freedom model which simulates the motion of two unit (single mass) vehicles in the X-Y plane. Thus, the vehicles move only in the X and Y directions and rotate only about their Z (yaw) axis. Any motion in the X-Z and Y-Z planes, or rotation about the x (roll) or y (pitch) axes is ignored.

Three routines are used for the basic vehicle model in EDSMAC. The first routine is called DAUX. It calls the TRAJ routine which computes tire forces and the COLL routine which computes collision forces. DAUX then uses these forces to calculate the linear and angular acceleration. The following steps summarize the procedures which occur at each time step (the name of the routine responsible for the calculations is shown in parentheses):

- compute the vertical wheel loads (INITIALIZE)

  NOTE: Since there are no load transfers in EDSMAC, this is done only once.

- assign the coefficient of friction according to the wheel location relative to the terrain boundary (TRAJ)
- compute the slip angle at each tire (TRAJ)
- compute the forward and lateral tire force at each wheel (TRAJ)
- sum the tire forces and moments (TRAJ)
- compute the forward and lateral collision force (COLL)
- sum the collision forces and moments (COLL)
- sum the total external forces and moments acting on the vehicle (DAUX)
- compute the resulting acceleration (DAUX)

Each of these tasks is described below.

Computing Vertical Wheel Loads

The free-body diagram shown in figure 12 illustrates the vehicle configuration and applied forces. The collision force and moment will be described shortly. For now, our concern lies with the tire force.

The vertical load at each wheel is simply the static load (load transfers are not included in EDSMAC). These static vertical loads are

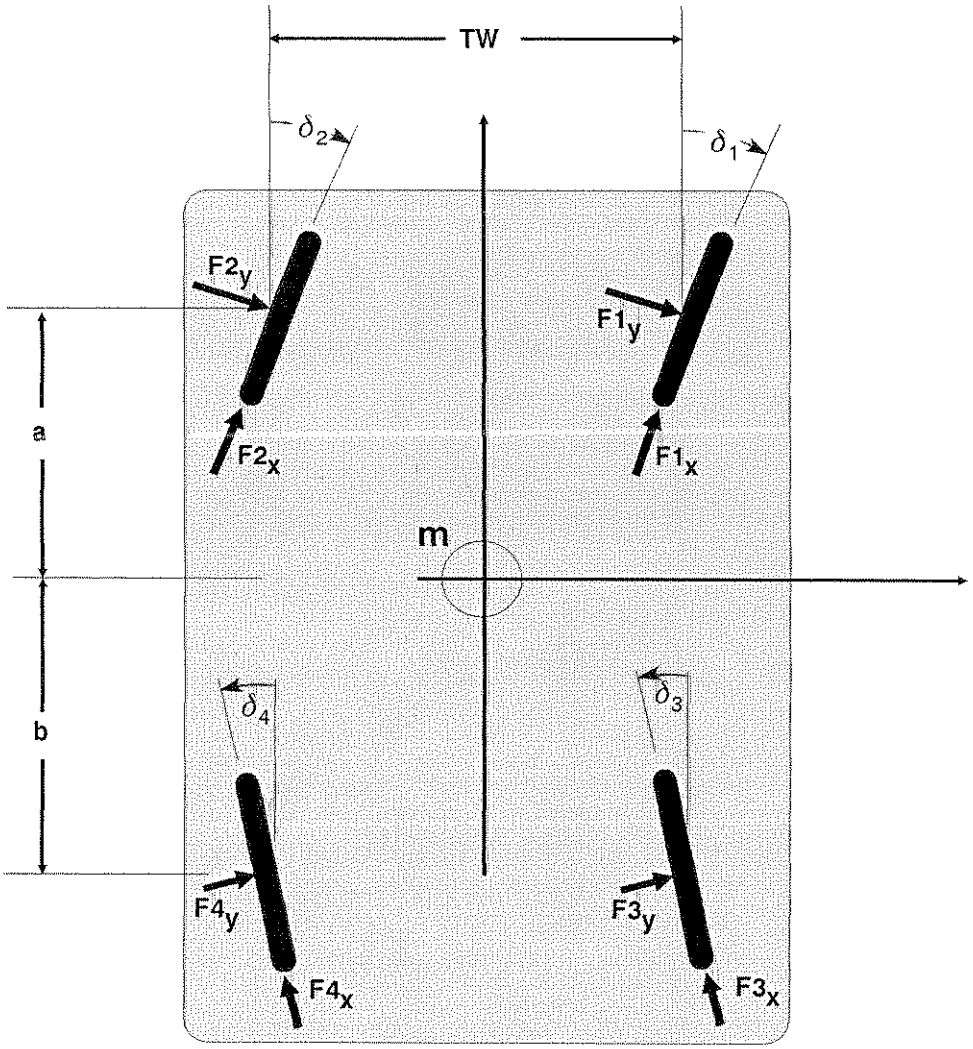

Figure 12 - Free-body Diagram of EDSMAC Vehicle Model

simply computed from the weight distribution by taking the sum of moments and forces in the Z direction:

$$F1_z = bW/(2(a + b))$$
$$F2_z = F1_z$$
$$F3_z = aW/(2(a + b))$$
$$F4_z = F3_z$$

These same calculations are performed for a second vehicle, if included in the analysis.

Assign the Coefficient of Friction

EDSMAC allows the user to specify a straight line, called a terrain boundary, which separates two zones having different coefficients of friction (see figure 13).

EDSMAC determines location of each wheel individually, and then assigns the appropriate coefficient of friction. The earth-fixed X,Y coordinates for each wheel are computed as follows:

Right Front Wheel:

$$X = X_{cg} + a\cos\psi - TW/2\sin\psi$$
$$Y = Y_{cg} + a\sin\psi + TW/2\cos\psi$$

Left Front Wheel:

$$X = X_{cg} + a\cos\psi - TW/2\sin\psi$$
$$Y = Y_{cg} + a\sin\psi + TW/2\cos\psi$$

Right Rear Wheel:

$$X = X_{cg} - b\cos\psi - TW/2\sin\psi$$
$$Y = Y_{cg} - b\sin\psi + TW/2\cos\psi$$

Left Rear Wheel:

$$X = X_{cg} - b\cos\psi - TW/2\sin\psi$$
$$Y = Y_{cg} - b\sin\psi + TW/2\cos\psi$$

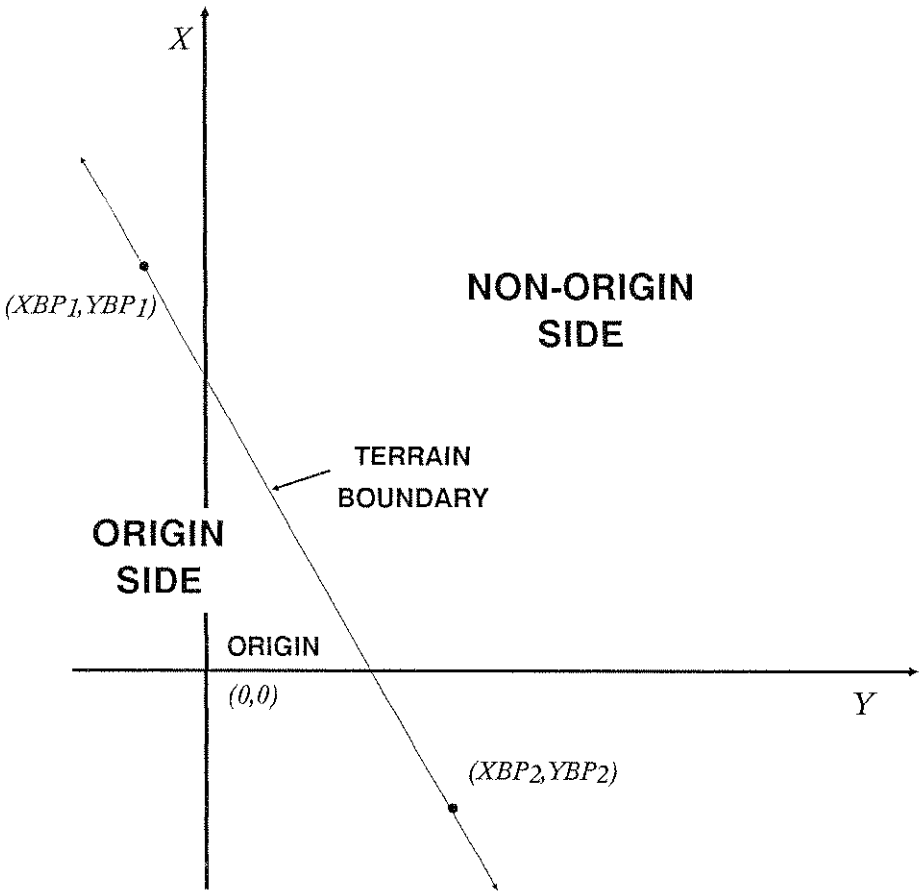

Figure 13 - Terrain Boundary

Now that the earth-fixed X,Y coordinates of each wheel are known, the next step is to determine on which side of the terrain boundary the wheel lies. The coefficient of friction is then assigned accordingly. The procedure uses the slope of the terrain boundary relative to the X axis, XSLOPE, and the Y intercept, YINT. The logic is performed as follows:

If XSLOPE = 0 degrees, the terrain boundary is parallel to the X axis. Under this condition, if the sign of ($Y_{boundary}$ - Y_{wheel}) is the same as the sign of Y_{wheel}, then the wheel lies on the non-origin side of the terrain boundary. Otherwise, the the wheel lies on the origin side.

If XSLOPE = 90 degrees, the terrain boundary is parallel to the X axis. Under this condition, if the sign of ($X_{boundary}$ - X_{wheel}) is the same as the sign of X_{wheel}, then the wheel lies on the non-origin side of the terrain boundary. Otherwise, the the wheel lies on the origin side.

Otherwise the terrain boundary is angled. If the sign of YINT is the same as the sign of (Y_{wheel} - XSLOPE*X_{wheel} - YINT), then the wheel lies on the non-origin side of the terrain boundary. Otherwise, the wheel lies on the origin side.

Computing Slip Angles

The relationship used for computing the slip angle at each tire is shown in figure 14. This relationship is based on vehicle geometry and kinematics, as well as the front axle steer angle, δ (which is obtained from the user-entered steer table). Slip angles for left side and right side tires are assumed to be equal - a good assumption as long as the track width is small compared to the turn radius. The slip angle for each tire is computed as follows:

Front tires -

$$\alpha_{1,2} = \text{ATAN2}((v + a\dot{\psi})/u) - \delta_{1,2}$$

Rear tires -

$$\alpha_{3,4} = \text{ATAN2}((v - b\dot{\psi})/u) - \delta_{3,4}$$

These tire slip angles and vertical loads are used to compute the forward and lateral tire forces, F_x and F_y. Unlike EDSVS and EDVTS, where the tire forces are computed in a separate routine, EDSMAC computes these tire forces in TRAJ, along with the tire slip angles and vertical loads (see **Tire Forces**).

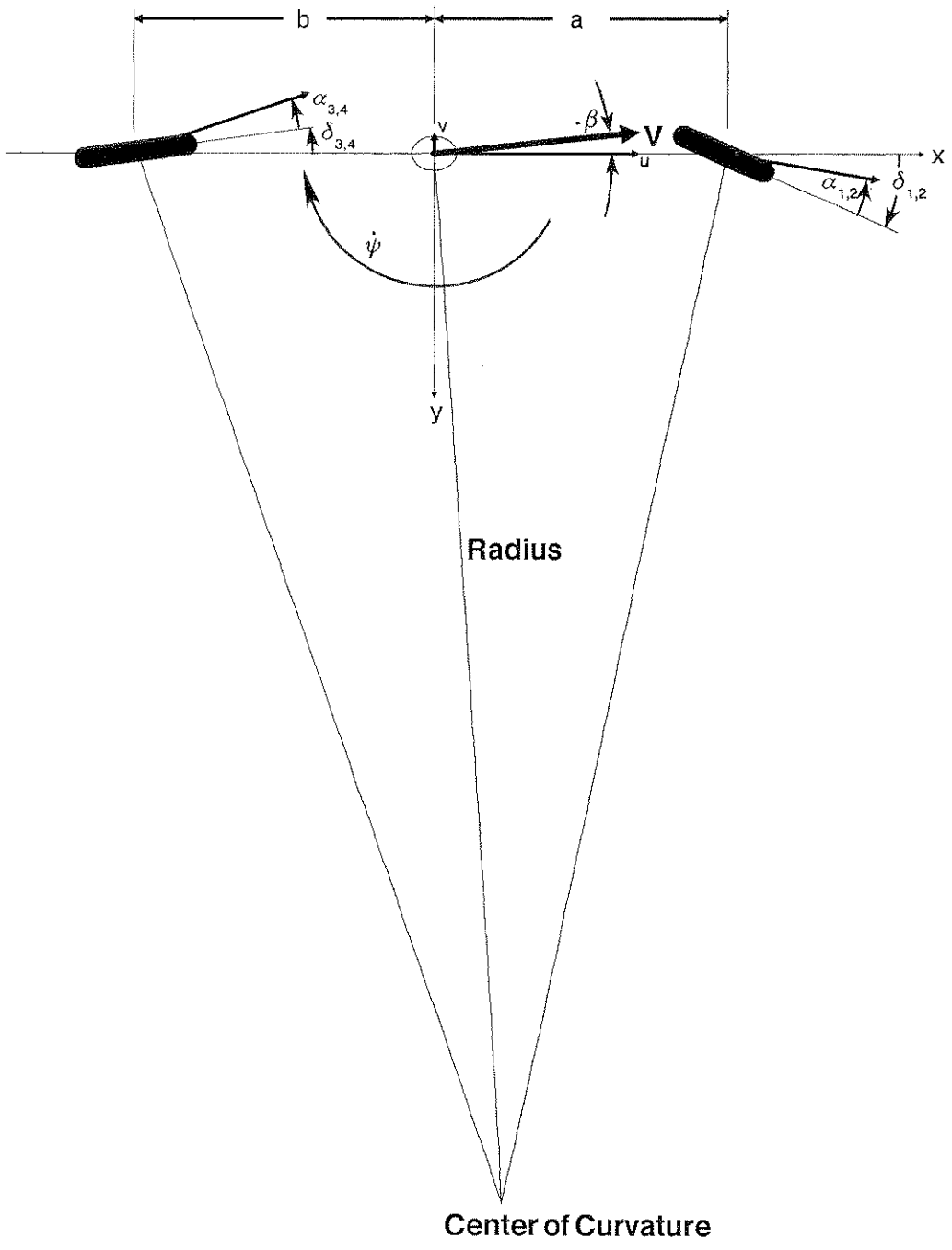

Figure 14 - Kinematic Relationship for Computing Slip Angles

Sum of the Tire Forces and Moments

The final step in the TRAJ routine is to sum the total tire forces and moments acting at the vehicle CG. This is performed as follows:

$$\Sigma F_x = F1_x\cos\delta_1 - F1_y\sin\delta_1 + F2_x\cos\delta_2 - F2_y\sin\delta_2 + F3_x\cos\delta_3 - F3_y\sin\delta_3 + F4_x\cos\delta_4 - F4_y\sin\delta_4$$

$$\Sigma F_y = F1_x\sin\delta_1 + F1_y\cos\delta_1 + F2_x\sin\delta_2 + F2_y\cos\delta_2 + F3_x\sin\delta_3 + F3_y\cos\delta_3 + F4_x\sin\delta_4 + F4_y\cos\delta_4$$

$$\begin{aligned}\Sigma M_z = {} & TW/2(-F1_x\cos\delta_1 + F1_y\sin\delta_1) + a(F1_x\sin\delta_1 + F1_y\cos\delta_1) \\ & + TW/2(F2_x\cos\delta_2 - F2_y\sin\delta_2) + a(F2_x\sin\delta_2 + F2_y\cos\delta_2) \\ & + TW/2(-F3_x\cos\delta_3 + F3_y\sin\delta_3) - b(F3_x\sin\delta_3 + F3_y\cos\delta_3) \\ & + TW/2(F4_x\cos\delta_4 - F4_y\sin\delta_4) - b(F4_x\sin\delta_4 + F4_y\cos\delta_4)\end{aligned}$$

Computing the Collision Forces and Moments

We have returned to the DAUX routine. Next, DAUX calls COLL to calculate the collision forces and resulting moments. The procedures are described in a later section (see **Collision Forces**) and associated references [15,16,17].

Sum of the External Forces and Moments

At a given timestep, the total forces and moments acting on the vehicle(s) are simply the sum of the tire and collision forces. DAUX performs these calculations. The total forces and moments acting at the vehicle CG are computed as follows:

$$\Sigma F_{x,total} = \Sigma F_{x,tires} + \Sigma F_{x,collision}$$

$$\Sigma F_{y,total} = \Sigma F_{y,tires} + \Sigma F_{y,collision}$$

$$\Sigma M_{z,total} = \Sigma M_{z,tires} + \Sigma M_{z,collision}$$

These calculations are performed for each vehicle.

Computing the Acceleration

The final step is to compute the accelerations (linear and angular) using Newton's 2nd law. These accelerations are derived with respect to the earth-fixed coordinate system. However, all the forces are with respect to the vehicle-fixed coordinate system. Thus, a transformation is required.

In the earth-fixed X direction, $\Sigma F_X = ma_X = m(\dot{u} - v\dot{\psi})$. Thus by rearranging, we obtain the acceleration in the X direction,

$$\dot{u} = \Sigma F_X/m + v\dot{\psi}$$

Similarly, in the Y direction we have $\Sigma F_Y = ma_Y = m(\dot{v} + u\dot{\psi})$. Thus,

$$\dot{v} = \Sigma F_Y/m - u\dot{\psi}$$

And about the Z axis, $\Sigma M_Z = I\alpha$ (note that angular acceleration is identical in both the earth-fixed and vehicle-fixed coordinate systems). Solving for angular acceleration yields

$$\alpha = \Sigma M_Z/I$$

These linear and angular accelerations are returned to the numerical integration routine for further processing to determine the velocities and positions at the next time step.

TIRE FORCES

Tire force calculations provide all the external forces acting on the vehicle in EDSVS and EDVTS (and EDSMAC during the non-collision phases). Since these forces are used to compute the resulting vehicle acceleration (which in turn, are used to calculate the velocities and positions), one can appreciate just how important the tire force calculations are.

This chapter begins by describing in general terms how these tire forces are computed for simulation programs. Then the specific tire models used by EDSVS, EDVTS and EDSMAC will be presented.

Basic Tire Mechanics

Tire mechanics is the study of how tires produce forces which act on a vehicle. As with the mechanics of any object, we will first define the coordinate system. To be consistent with the Society of Automotive Engineers (and all EDVAP programs), we will use SAE Recommended Practice SAE-J670e [18] (included in your EDVAP Appendix). The tire coordinate system is shown on the following page.

The coordinate system shows forces produced in the X', Y', and Z' directions and moments produced about the X', Y' and Z' axes. However, for a yaw plane analysis, all forces are assumed to be produced at the ground plane (X'-Y'). We can further choose to ignore any aligning torque and the moments produced about the Z' axis. This greatly simplifies the scope of our study. We now need only consider only forces produced in the X' (forward) and Y' (lateral) directions.

NOTE: The X'-Y' axes are defined relative to the tire, *not* the vehicle!

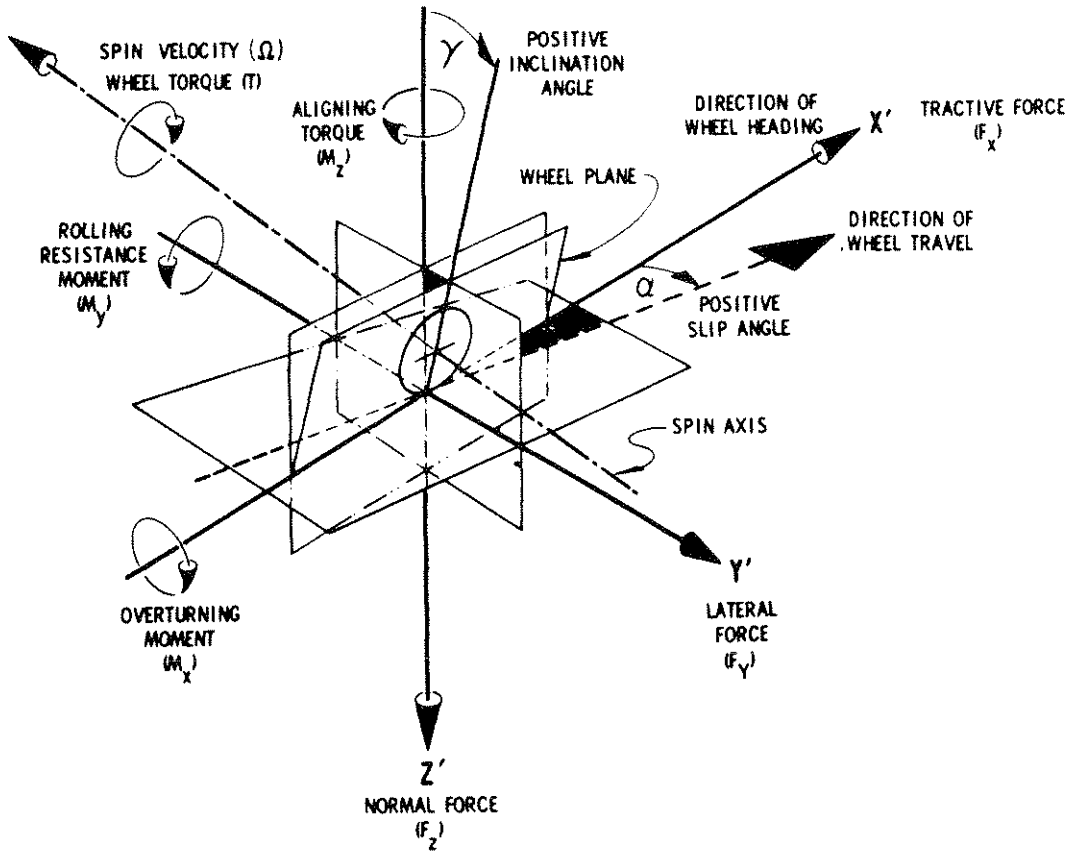

Figure 15 - Tire Coordinate Axis System (from Ref [18])

The portion of the tire tread in contact with the ground is determined by the inflation pressure and tire load. This area is equal to the vertical load divided by inflation pressure.

This area is called the tire contact patch. In general, this area is approximately defined by the product of the tread width and the tread length in contact with the ground. Since the width of the tread surface is fixed, the length of the contact patch is load divided by the product of pressure and tread width)

The contact patch is also called the tire-road shear interface. This term is descriptive of the actual process by which tire forces are produced. As shown in figure 16, the forces produced between the

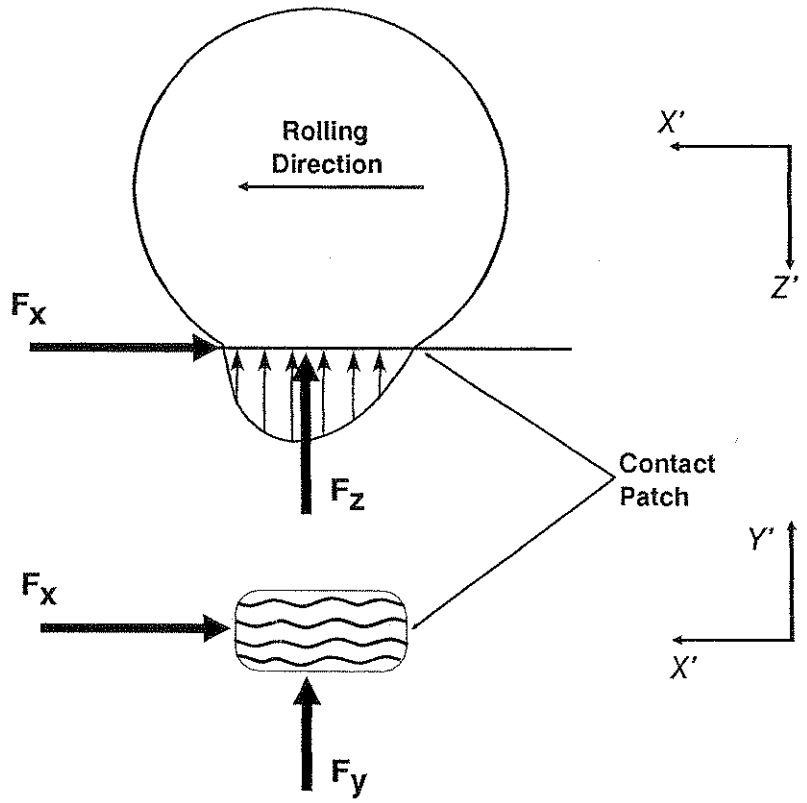

Figure 16 - Forces at the Tire Contact Patch

tire and road are shear forces which are the result of the tire pressure distribution and the coefficient of friction. Clearly, the shape of the pressure distribution determines the location of the net effective tire force. This factor is important when analyzing tire aligning torque and overturning moments. However, these factors are beyond our needs. The interested reader is referred to the literature [19].

Our concern is how to estimate the overall longitudinal and lateral shear forces, F_x and F_y, produced by the tire. This process is described next.

Longitudinal Tire Force

A tire produces a longitudinal force at the contact patch caused by braking and accelerating torques applied to the axle. These forces are normally much greater than the small external rolling resistance force applied at the contact patch as shown in figure 15.

Longitudinal tire force is rather easy to estimate; it is equal to the summation of the applied forces (rolling resistance plus tractive or braking force), and is limited by the available friction.

Longitudinal tire force is accompanied by tire slip. As the force increases, the slip increases. This continues in a rather linear fashion until the friction limit is reached. This friction limit is called the peak friction coefficient, μ_p. Any additional attempted force causes the wheel to lock (100 percent slip) and the available friction drops to the locked wheel (or slide) friction coefficient, μ_s.

A typical graph of longitudinal force vs slip is shown in figure 17.

Lateral Tire Force

A free-rolling (i.e., no longitudinal tire force) tire produces lateral force when the rolling direction of the tire is not in the tire plane. The angle between the rolling direction and the tire plane is a key parameter called the tire *slip angle*, α. Measurements have shown that for small slip angles,

$$F_y = C_\alpha \alpha$$

where

F_y = lateral tire force
C_α = Cornering stiffness
α = slip angle

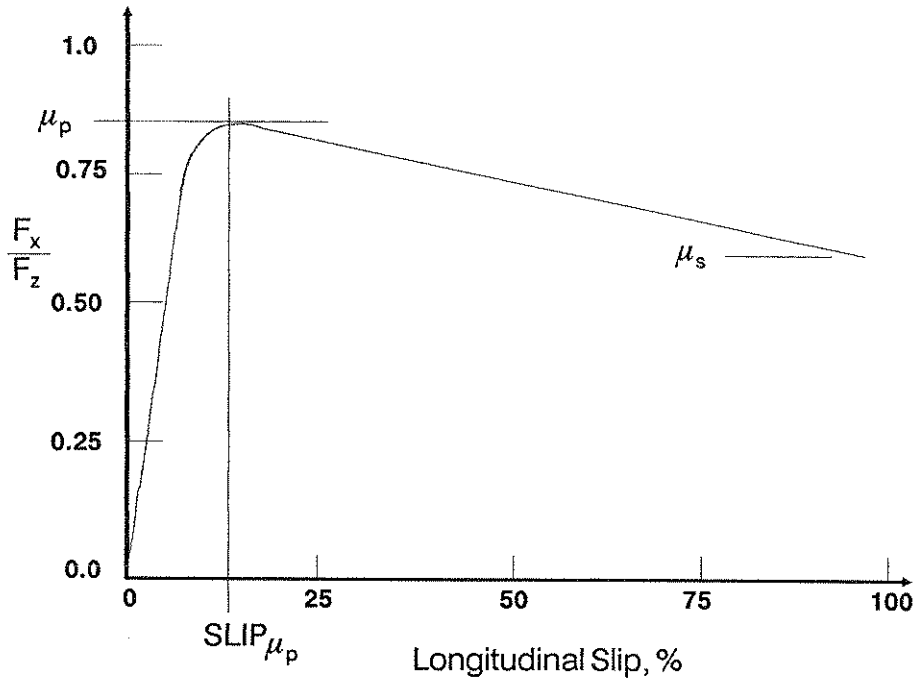

Figure 17 - Typical Longitudinal Force vs Slip Curve

However, as the slip angle increases the force vs slip angle relationship becomes non-linear (see figure 18). For this reason, a *tire model* is used to calculate lateral force. The Fiala tire model [20] is one such model which has been used successfully by several vehicle simulations, including those developed at Calspan [3,5] and The University of Michigan [7]. The Fiala tire model is also used by EDSVS, EDVTS and EDSMAC.

The Fiala model is based on the non-dimensional parameter, $\bar{\alpha}$,

$$\bar{\alpha} = \frac{C_\alpha \tan\alpha}{\mu F_z}$$

where

μ = Coefficient of friction
F_z = Vertical tire load

By Fiala's analysis, the lateral force is equal to

$$F_y = \mu F_z[-\bar{\alpha} + \bar{\alpha}^2/3 + \bar{\alpha}^3/27] \qquad \text{for } |\bar{\alpha}| < 3$$

and

$$F_y = \mu F_z \sin\alpha \qquad \text{for } |\bar{\alpha}| \geq 3$$

Figure 18 compares Fiala's relationship for typical car and truck tires. As is evident from the comparison, Fiala's theoretical predictions match measured values quite well.

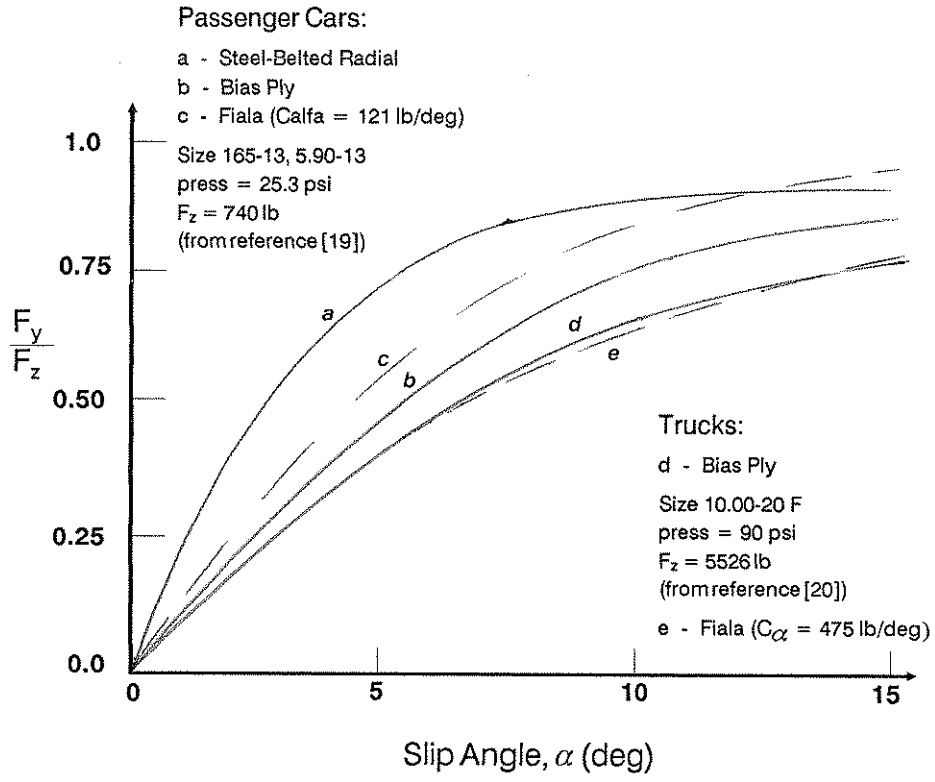

Figure 18 - Comparison Between Predicted and Measured Values

Combined Longitudinal and Lateral Force

Under the conditions of combined braking and steering, a tire produces longitudinal and lateral forces simultaneously. Experiments have shown that conditions of lateral slip reduce the longitudinal force and vice versa.

Several methods have been employed in various vehicle simulation programs which model this phenomenon. Two popular methods are

- friction circle
- slip vs rolloff table

Each of these methods is descibed below.

The Friction Circle

In the ground plane, tire forces are produced in the X and Y directions. These forces are limited by the available friction force, μF_Z, where μ is the coefficient of friction between the tire and roadway, and F_Z is the vertical tire force. This observation has lead to the following general relationship between forward and lateral forces and the total available force:

$$\sqrt{F_x^2 + F_y^2} \leq \mu F_z$$

This is simply the equation of a circle which has its origin at (0,0). This relationship between forward, lateral and total available tire force is called the *friction circle*. As shown on the following page (see figure 19), the friction circle is a powerful way of visualizing the possible range of lateral (steering) force given that a certain amount of braking exists.

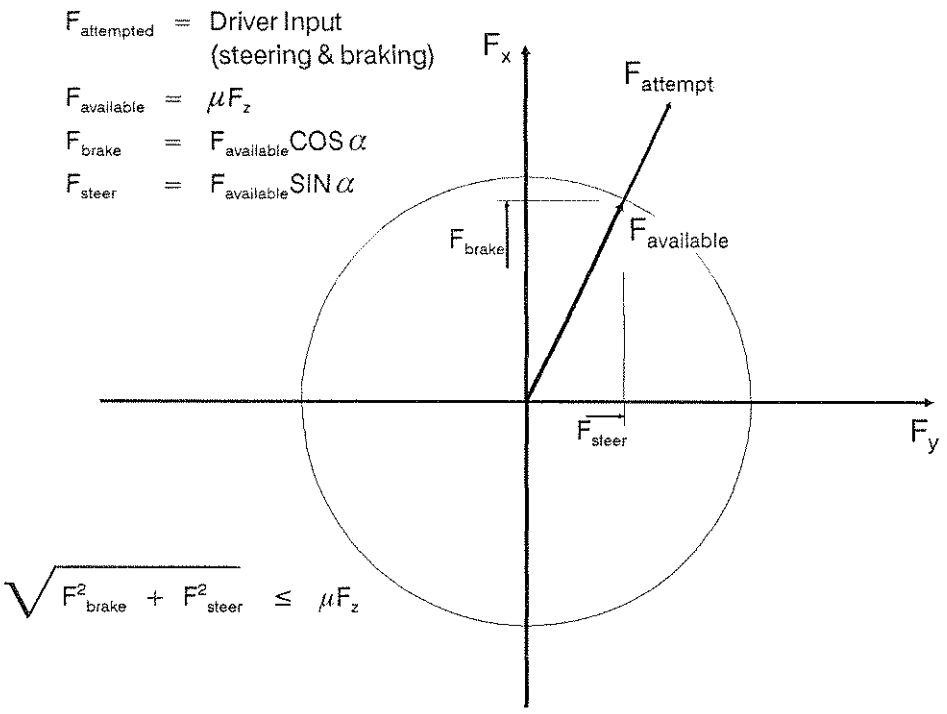

Figure 19 - Friction Circle

EDSMAC uses the friction circle to determine how much lateral tire force can be developed, given the current level of braking and the available friction. Referring to figure 19, the available force is the radius of the circle, μF_Z. Let F_{attemp} be the attempted braking force. Note that $F_{attempt}$ must be equal to or less than μF_Z. The braking and steering tire forces are $F\cos\alpha$ and $F\sin\alpha$, respectively.

The Fiala tire model is then applied to determine if the computed lateral force is further modified (see Fiala, above).

Slip vs Rolloff

Another way of computing the loss of available lateral tire force at a given level of braking is the use of a *slip vs rolloff table*. This approach uses test data rather than an assumed friction circle

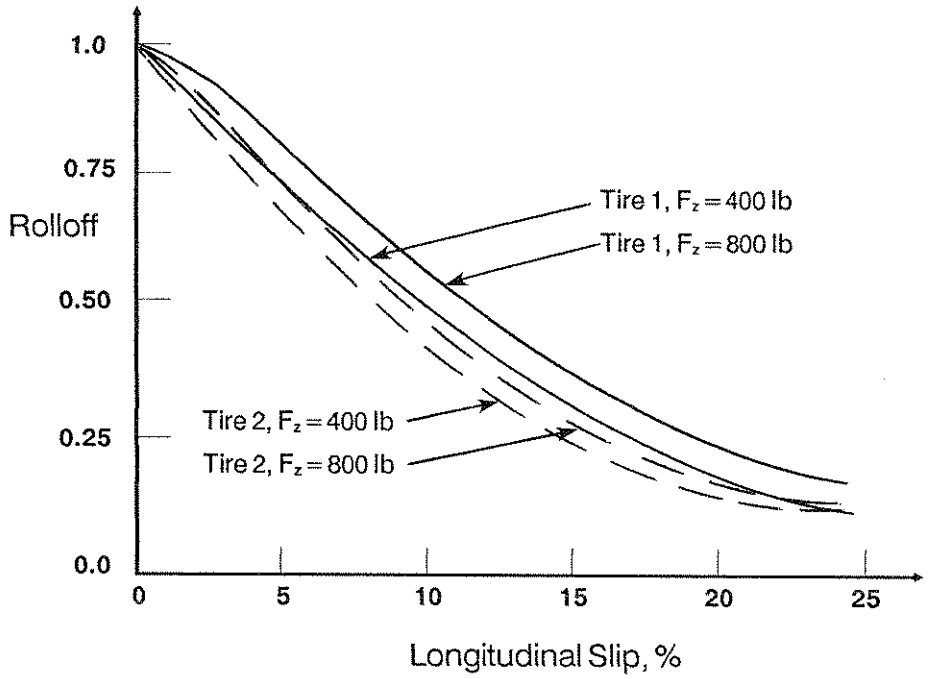

Figure 20 - Slip vs Rolloff

relationship. This data is produced by placing a tire in a test machine which subjects the tire to a measureable amount of longitudinal tire slip. The loss in maximum lateral force generation is noted. The result is a slip vs rolloff table for the test tire. Typical examples of slip vs rolloff curves are shown in figure 20 [21]. The slip vs rolloff relationship is rather insensitive to speed, load and tire construction (radial vs bias).

EDSVS and EDVTS use a slip vs rolloff table to determine how much lateral tire force can be developed, given the current level of tire slip. This is accomplished as follows:

First, compute the maximum lateral tire force based on the Fiala model:

$$\overline{\alpha} = C_\alpha \alpha / \mu_p F_Z$$

if $\bar{\alpha} < 3$ then

$$F_y = -\mu_p F_z(-\bar{\alpha} + \bar{\alpha}^2/3 - \bar{\alpha}^3/27)$$

otherwise

$$F_y = -\mu_p F_z$$

If the tire is sideslipping ($\alpha \neq 0$), reduce the peak longitudinal friction coefficient, μ_p, according to the following empirical formula

$$\mu_p = \mu_p(1-1.72\alpha)$$

μ_p is not allowed to exceed the slide coefficient of friction,

$$\mu_p \leq \mu_s \cos \alpha$$

If the attempted longitudinal force is less than the available force, then perform the following steps:

- Compute the percent tire slip. Letting s equal the percent of wheel lockup associated with μ_p, the tire slip is

  $$\text{slip} = ((F_{att}/F_z)/\mu_p)*s$$

- Use the slip vs rolloff curve to determine the rolloff at the specified slip

  $$f(s) = \text{function of slip vs rolloff table (figure 20)}$$

- Compute the reduced lateral force, F_y

  $$F_y = F_y * f(s)$$

Otherwise, the attempted longitudinal force is greater than the available force, so assign the forward and lateral components limited by the slide coefficient of friction, μ_s

$$F_x = \mu_s F_z \cos \alpha$$
$$F_y = \mu_s F_z \sin \alpha$$

Note that the above condition is analogous to the friction circle since

$$\sqrt{F_x^2 + F_y^2} = 1$$

COLLISION FORCES

The COLL routine computes the inter-vehicle force at discrete time intervals specified by the numerical integration process. The resulting change in velocity for each successive time increment are later summed (in the SAVEMAX routine) to provide the delta-V for each vehicle during the collision phase. This chapter describes the inner workings of the COLL routine.

Computing the Inter-vehicle Force

The flow chart in figure 21 describes the procedure used by the COLL routine. The process takes many steps. The majority of the task is spent simply in keeping track of various angles and collision vectors. Each of these steps is described below.

Setting up the Relative Coordinates w/ Veh #1 as Base

Vehicle positions (X and Y center of gravity (CG) coordinates and heading angle) are entered by the user and continuously updated by numerical integration as the vehicles move in the earth-fixed coordinate system. At each time increment, a vehicle-fixed coordinate system is established, first using vehicle no. 1 as the point of reference (see figure 22). At this point in time, a set of indices, *I* and *J*, are used to describe the point of reference. The index, I, identifies the *base* vehicle (i.e., vehicle 1) and the index, J, identifies the *other* vehicle (i.e., vehicle 2).

As shown in figure 22, vehicle no. 2 can be precisely located from vehicle no. 1's reference frame by using vehicle no. 2's current earth-fixed X,Y coordinates and heading angle. Because vehicle no. 2's exterior dimensions are also known, two angles, PSIBPB and PSIBPF can be established. Logically, any damage to the surface of the base vehicle (in this case, vehicle 1) must occur between these two angles.

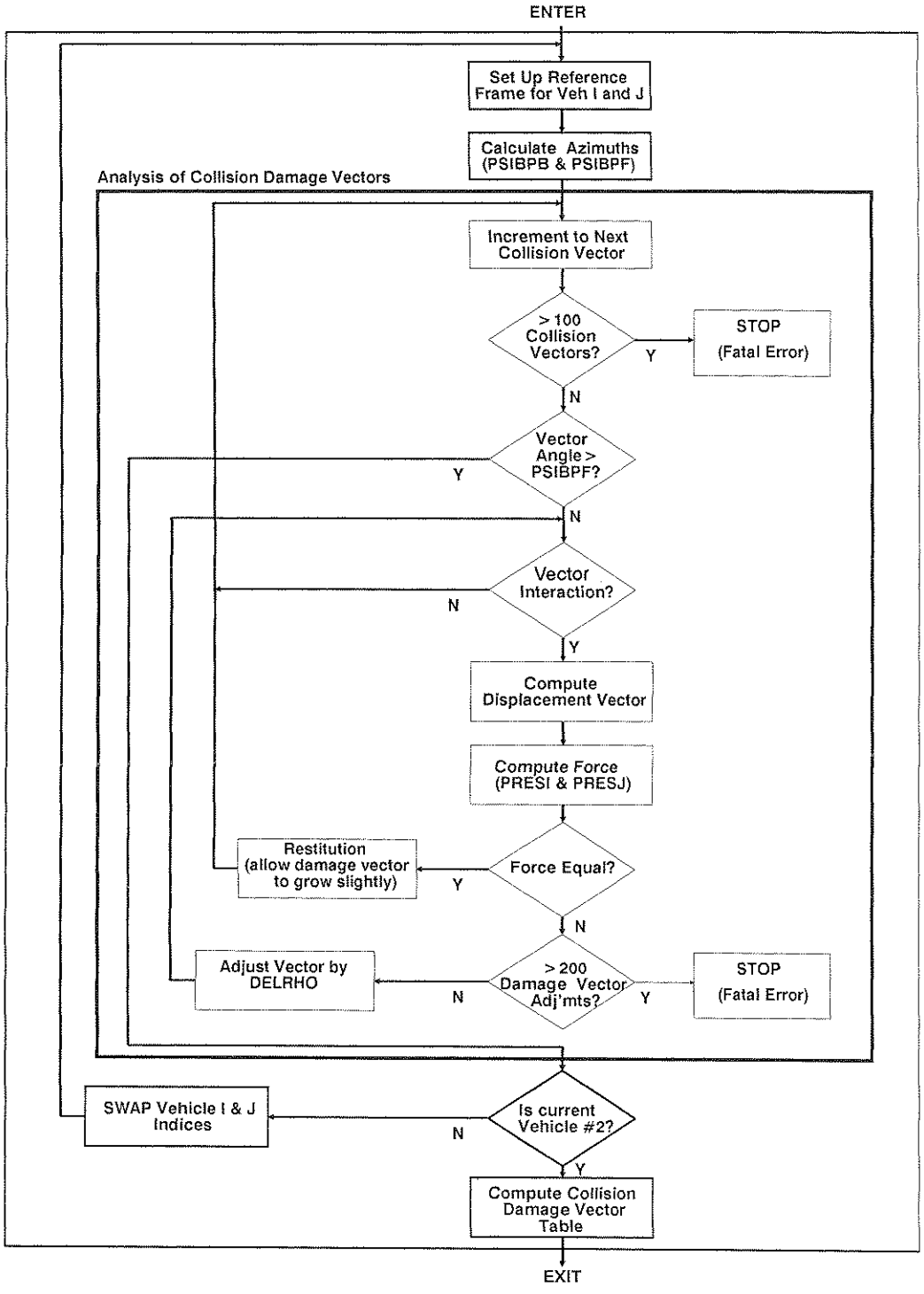

Figure 21 - EDSMAC Collision Force Routine, COLL

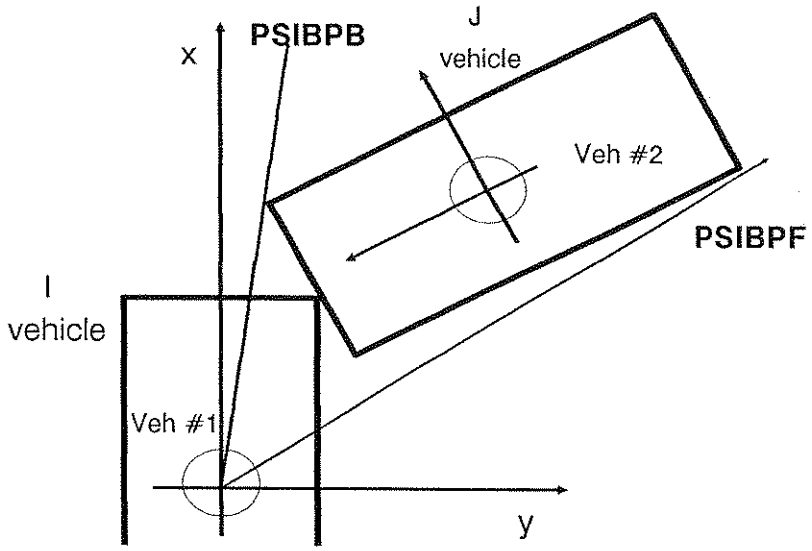

Figure 22 - Viewing vehicle no. 2 from vehicle no. 1's reference frame. Note azimuth angles, PSIBPB and PSIBPF, which define the maximum possible range of damage.

Calculating the Radial Vectors

A set of equally spaced radial vectors is next established (see figure 23). Each vector, RHO, originates at the CG of the base vehicle and extends towards the other vehicle within the range between PSIBPB and PSIBPF. Each RHO vector represents a potential force against the vehicle. The angle between each vector on the base vehicle is established by the value of DELPSI, an input variable (usually about 2 degrees). The angle on the other vehicle is almost always different than the base vehicle because, while the vector endpoints are the same for both vehicles, the distances to the CGs are usually not the same.

Seeking Interaction

Interaction (i.e., a collision) between the vehicles is confirmed when a point on the exterior of the base vehicle lies within the perimeter of the other vehicle. With assistance from figure 24, it

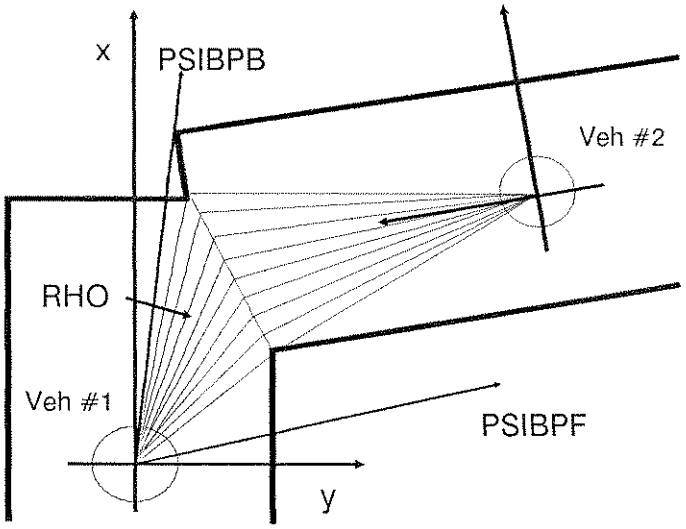

Figure 23 - RHO vectors from base vehicle within damage range defined by PSIBPB and PSIBPF. Vectors are equally spaced DELPSI degrees.

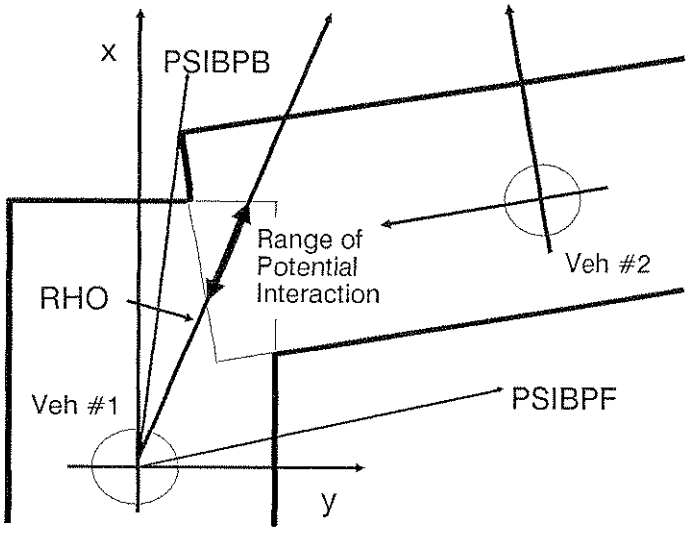

Figure 24 - A collision is confirmed by seeking interaction along a particular RHO vector, i.e., a point on the vector lies within the perimeter of both vehicles.

can be seen that interaction along a particular RHO vector exists if a point on that vector lies within the perimeter of both vehicles. This test is performed for each vector within the potential interaction range from PSIBPB to PSIBPF. Points which satisfy this test are called interaction points and provide the basis for what will become the damage profile.The perimeter of each vehicle changes as it is damaged. Therefore, the perimeter is saved at the end of each collision time step. This prevents sensing a "false interaction" at areas which no longer exist because of prior damage.

Computing the Displacement Vector

Once interaction for a particular RHO vector is confirmed, two additional vectors are established (see figure 25). RHOBI is a vector from the base vehicle CG to the interaction point and RHOBJ is a vector from the other vehicle CG to the interaction point. Note the use of I and J for the base vehicle and other vehicle, respectively. This naming convention is used consistently when identifying vectors and angles on the base and other vehicle. The distance from the previous vehicle perimeter to a new interaction point specifies a change in the length of the RHOBI and RHOBJ vectors.This interaction point represents a location on the vehicle where the inter-vehicle force is computed. The force at each interaction point is equal to

```
PRESI  = AKVI*DELTAI        (base vehicle)
```

and

```
PRESJ  = AKVJ*DELTAJ        (other vehicle)
```

where DELTAI and DELTAJ are the distances from the original vehicle perimeters to the end of the RHOBI and RHOBJ vectors, respectively. Likewise, AKVI and AKVJ are the crush stiffness coefficients for vehicles 1 and 2, respectively. AKVI and AKVJ are input quantities, usually in the range of 40 to 100 lb/in per inch of damage width (or lb/in^2). As shown in figure 25, PRESI and

PRESJ are not necessarily in the directions of the RHOBI and RHOBJ vectors; PRESI and PRESJ must be mutually opposing to correctly satisfy Newton's 3rd law.
The next step is to compute the difference between PRESI and PRESJ. Again by Newton's 3rd law, PRESI and PRESJ should be equal; thus their difference should be zero. However, since the two forces are computed from separate information, there is little chance the computed forces will be equal. Instead, the force difference is compared to an allowable force difference, ALAMB, an input quantity (usually about 20 lb.). If the difference is less than ALAMB, then PRESI is set equal to PRESJ.
If the difference in PRESI and PRESJ is more than ALAMB, the interaction point is moved along the RHO vector by an increment of DELRHO (an input quantity, usually about 0.2 inches). From the new interaction point, DELTAI and DELTAJ are recomputed. A new set of PRESI and PRESJ values are then computed and compared with ALAMB. The interaction point for each RHO vector can be moved up to 200 times in an effort to equalize PRESI and PRESJ. If more than 200 adjustments occur, a fatal error is issued and calculations stop.

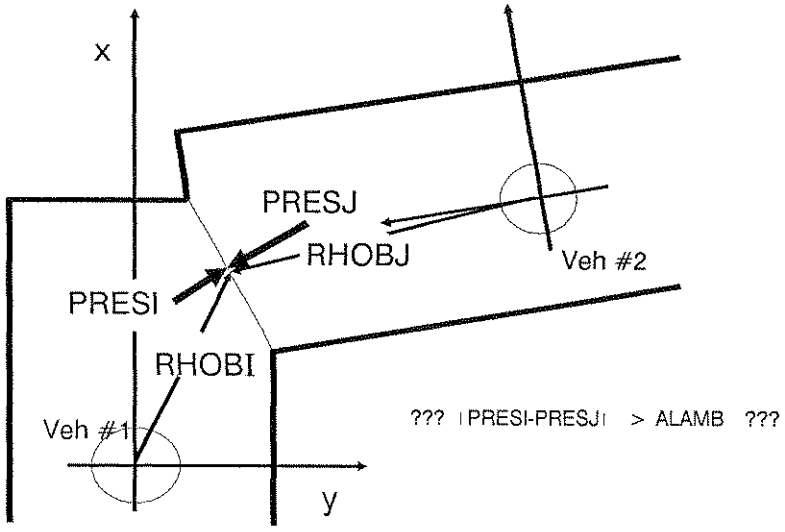

Figure 25 - Computing the displacement vectors. The displacement from the original surface is used to compute the collision force at each interaction point for both vehicles.

After PRESI and PRESJ are equalized, the next RHO vector is analyzed using the above process.

Up to 100 RHO vectors are allowed, resulting in a table for the base vehicle which includes the following results: the vehicle-fixed X,Y coordinates of the interaction point, the angle, PSIB, of the RHO vector, and the contact force, PRESI, at the interaction point. This table represents a damage profile shared between the vehicles.

It is possible for an interaction point for the base vehicle to be computed from the other vehicle's CG. These points on the damage profile are called *J-points* because they are associated with vehicle index J. These RHO vectors are flagged by double asterisks in the EDSMAC output (refer to the **Vehicle Damage Ranges**).

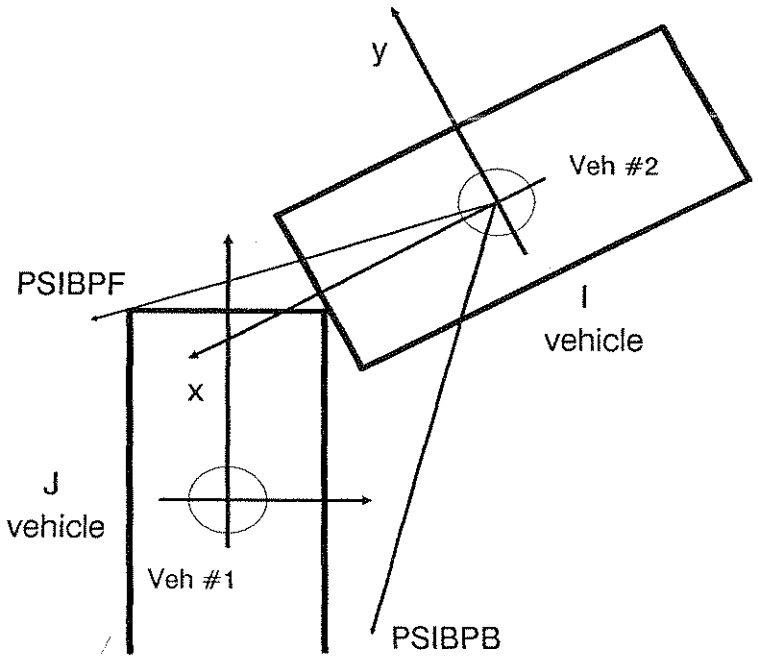

Figure 26 - After calculating from vehicle no. 1's reference frame, the vehicle indices, I and J, are swapped and calculations are repeated from vehicle no. 2's reference frame.

Restitution

Elastic restitution is modeled by allowing the length of the RHO vector to increase slightly from its equilibrium length. The method uses a second order polynomial. A program variable, C, is computed as follows:

$$C = C_0 - C_1\delta + C_2\delta^2 \qquad \text{for } 0 \le \delta < C_1/2C_2$$

$$= 0 \qquad \text{for } \delta > C_1/2C_2$$

where

C = Program variable associated with restitution ε, by the formula, $C = \sqrt{1 - \varepsilon^2}$

C_0, C_1, C_2, = Polynomial coefficients (user-input)

δ = Maximum deflection of a RHO vector

Swap and Repeat

The above calculations use vehicle no. 1 as the base vehicle. The next step is to swap reference frames and repeat the calculations, i.e., vehicle no. 2 is used as the *base* vehicle, *I*, and vehicle no. 1 is the *other* vehicle, *J*, as shown in figure 26 (preceding page).

Computing the Collision Force

The damage table provides up to 100 individual force vectors applied to each vehicle mass at a known location on the damage profile during the current time step. The final procedure is to use this information to compute the force normal (perpendicular) to the surface of the damage profile. The inter-vehicle friction factor, AMU, (an input quantity, usually about 0.30 to 0.75), is used to determine the maximum force acting tangentially on the damage surface. Finally, the sum of these forces is used to compute the total collision force and moment acting on the vehicle CG (see figure 27) during a given integration time step. These forces and moments are:

SFXC - sum of F_x from collision
SFYC - sum of F_y from collision
SFNC - sum of moments about z-axis

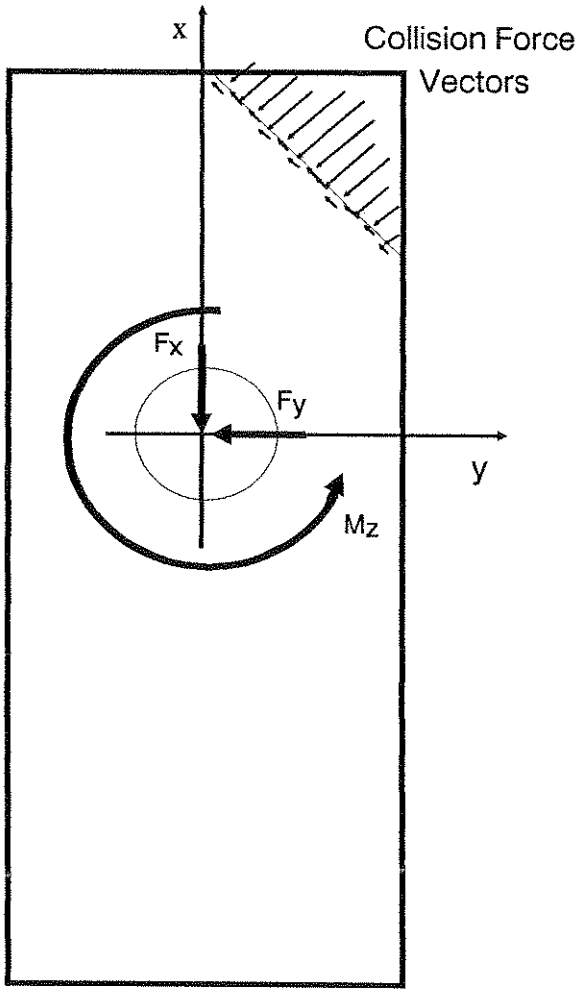

Figure 27 - Sum of forces and moments resulting from collision forces

Computing the Delta-V

From basic physics, it is known that dV = adt. Graphically, this is simply the area under the acceleration vs. time curve. This incremental area is the delta-V for the current time step. During the collision phase, the incremental areas for each time step are summed by using the average acceleration for the current time increment (see figure 28).

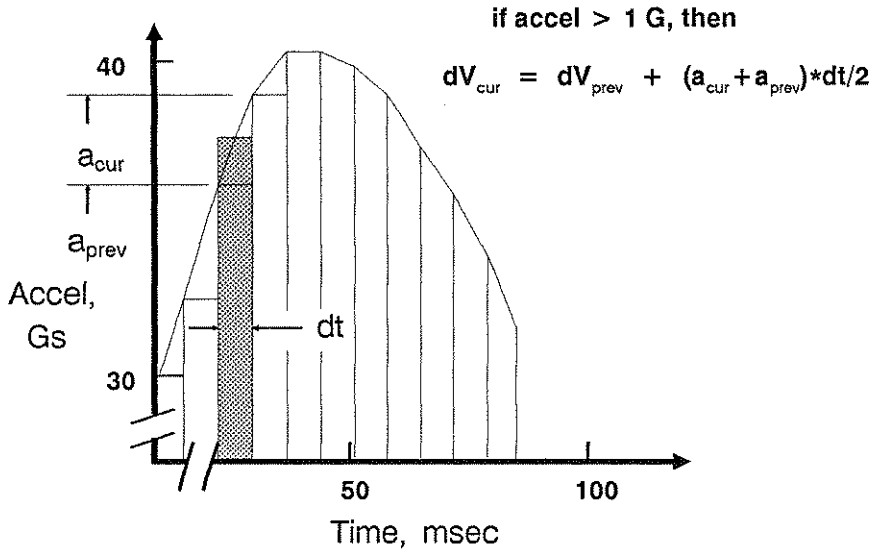

Figure 28 - The delta-V is computed from the total area under the acceleration vs. time curve. This process occurs during the SAVEMAX routine at each integration timestep.

At the conclusion of each integration time step, control is returned to the Main Loop. A subroutine named SAVEMAX in the main loop stores and accumulates the delta-V for display during output.

PDOF

The vehicle-fixed forward and lateral components of the acceleration vs. time history are also saved in subroutine SAVEMAX. These components define the direction of the delta-V. By definition, the direction of the delta-V vector is the Principal Direction of Force (PDOF). This direction is converted into an hour angle (12-o'clock through 11-o'clock) and used for assigning the first two characters of the CDC.

If the collision results in more than one CDC, each CDC will be reported in the EDSMAC output and graphics separately (see **Vehicle Damage Ranges** and **Damage Profiles**, respectively. Each individual CDC may also have a different delta-Vs. However, this only occurs if the individual CDCs have different PDOF clock directions.

NUMERICAL INTEGRATION

Numerical integration is the process of obtaining an approximate numerical solution of a system of ordinary differential equations. In our case, the differential equations are the equations of motion,

$$dv/dt = a = f(t,v(t))$$
$$ds/dt = v = f(t,s(t))$$

with the known initial values (velocity and position),

$$v(t_o) = v_o$$
$$s(t_o) = s_o$$

Given this situation, we seek the values

$$v(t_o + \Delta t)$$
$$s(t_o + \Delta t)$$

where Δt is a small increment of time, t.

Runge-Kutta Method

The Runge-Kutta method of numerical integration was first proposed by C. Runge, a German mathematician and physist, in 1895 [22], and later modified by M. Kutta in 1901 [23]. The algorithm approximates the Taylor series,

$$y(x_o + \Delta X) = y(x_o) + \Delta x y'(x_o) + \Delta x^2/2y''(x_o) + \ldots + \Delta x^n/n!y^n(x_o)$$

As shown above the Taylor series requires up to the n^{th} derivative to be calculated. Only the first derivative is required by the Runge-Kutta method. However, this derivative must be evaluated four times to insure sufficient accuracy.

Flow Chart

A complete mathematical development of the Runge-Kutta method is presented in the literature [24,25,26] and is beyond our scope. However, a review of the calculation steps is useful. A flow chart illustrating these steps is shown below.

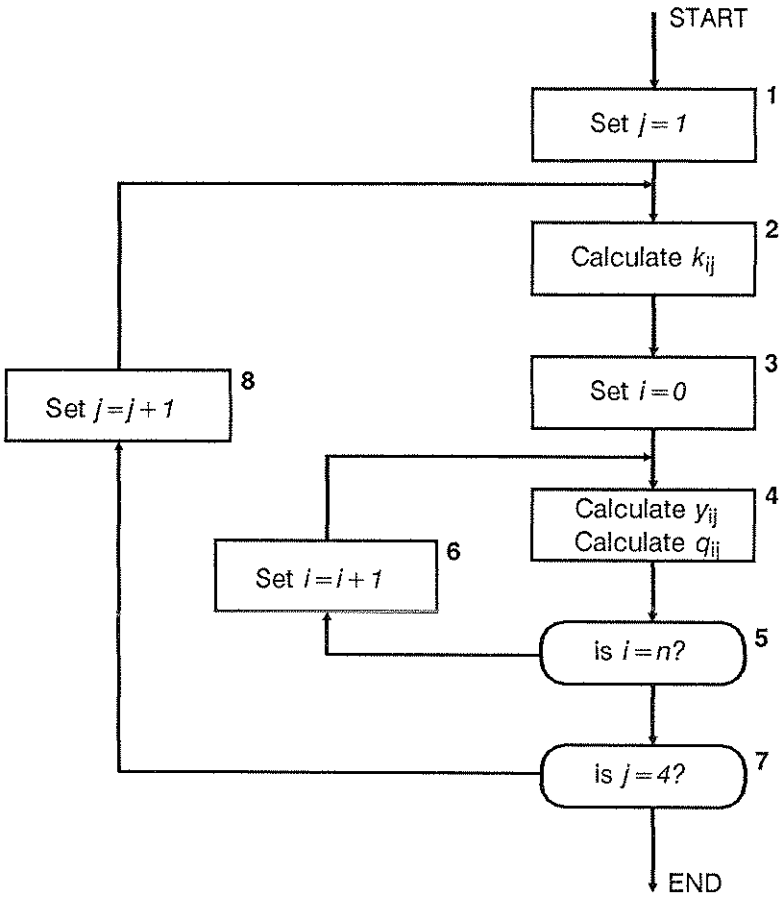

Figure 29- Runge-Kutta Numerical Integration Flow Chart [24]

In the above flow chart,

n = the number of 1st order differential equations
i = index for the ith equation
j = index for jth iteration
k = derivative vector (computed externally)
y = solution vector
q = auxilliary storage vector of coefficients (internal use only)

Each step is described as follows:

Step 1 - Initialize the interation index, j

Step 2 - Calculate the derivative vector for up to n equations

Step 3 - Initialize the equation index, i

Step 4 - Calculate the i^{th} solution vector, k_{ij}, for the j^{th} iteration

Step 5 - Decide whether all equations in the solution vector have been calculated

Step 6 - If necessary, increment theequation index, i

Step 7 - Decide if the calculation is complete (i.e., $j = 4$). If so, return control to the calling program.

Step 8 - If necessary, increment the iteration index, j, and return to step 2.

Notice that the derivative vector is computed four times (i.e., step 2 is repeated for j = 1,2,..,4). This increases the calculation time significantly since step 2 is where all the equations of motion are programmed. This is required for sufficient accuracy.

Application to EDSMAC

EDSMAC uses a 4th order Runge-Kutta method called RNGKT1. [27]. This routine was developed in-house at Cornell Aeronautical Lab. The code for RNGKT1 follows:

```
' -- Perform the first-order approximation of the solution vector, Y
FOR I = 1 TO N                  ' -- For each equation
    K(I) = DT*DER(I)            ' -- Estimate the next derivative
    Y(I) = Y(I) + .5*K(I)       ' -- Estimate the solution
    VAR(I) = Y(I)               ' -- Store the solution
    Q(I) = K(I)                 ' -- Store the first-order derivative
NEXT I

' -- Perform the second-order approximation
T = X + .5*DT                   ' -- Increment the simulation time
                                   by 1/2 of the time step
```

```
CALL DAUX                       ' -- Update the derivatives by
                                   performing the physics at the
                                   new simulation time, T, and
                                   position and velocity, VAR().
FOR I = 1 TO N
      K(I) = DT*DER(I)
      Y(I) = Y(I) + .5*(K(I) - Q(I))
      VAR(I) = Y(I)
      Q(I) = Q(I)/6
NEXT I
' -- Perform the third-order approximation
CALL DAUX                       ' -- Update the derivatives at the
                                   new position and velocity, VAR().
FOR I = 1 TO N
      K(I) = DT*DER(I) - .5*K(I)
      Y(I) = Y(I) + K(I)
      VAR(I) = Y(I)
      Q(I) = Q(I) - K(I)
NEXT I
' -- Perform the fourth-order approximation
T = X + DT                      ' -- Increment the simulation time
                                   by 1 full time step
CALL DAUX                       ' -- Update the derivatives  at the
                                   new time, T, and new
                                   position and velocity, VAR().
FOR I = 1 TO N
      K(I) = DT*DER(I) - .5*K(I)
      Y(I) = Y(I) + K(I)
      VAR(I) = Y(I)
      Q(I) = Q(I) - K(I)
NEXT I
CALL DAUX                       ' -- Update the derivatives  a final
                                   time at the new position and
                                   velocity, VAR().
END
```

At the conclusion of RNGKT1, EDSMAC returns to the controlling program which decides, among other things, if it is time to print, change the time step or end the simulation.

Predictor-Corrector Method

The Runge-Kutta method has two disadvantages. First, there is no evaluation of the error produced at each time step. Although this error tends to be quite small (on the order of Δt^5), it can become significant for some dynamic systems. Second, the step size is constant and the derivatives are re-evaluated four times, even if only one or two evaluations meet the required accuracy.

The predictor-corrector method of numerical integration overcomes both of these limitations. This method is similar to the Runge-Kutta method in that it is a 4^{th} order method based on the Taylor's series expansion. The predictor-corrector method estimates the error at each time step by comparing the current solution with the solution which was predicted during the evaluations at previous time steps. Usually, the results from the previous four time steps are saved for this purpose. When the difference between the predicted value and the current solution exceeds a predetermined amount, the time step is halved, a correction factor is applied, and a new solution is obtained. When the error is small for successive time steps, the time step is doubled. Thus, the predictor-corrector method can be more accurate while saving calculation time as well.

The disadvantage of the predictor-corrector method is that it is not *self-starting*. That is, the method cannot be used for the first four time steps because the *predictors* are not known. For this reason, a starting method is required. HPCG uses the 4^{th} order Runge-Kutta method for the first four time steps.

If all this seems fairly complicated, you are correct. While the Runge-Kutta method can be programed in approximately 25 lines of code, a predictor-corrector method requires well over 300 lines.

Application to EDSVS and EDVTS

EDSVS and EDVTS use a method of numerical integration called Hamming's Modified Predictor-Corrector Method, or HPCG [28].

For more information on the HPCG numerical integration routine, the interested reader is referred to Hamming's theoretical development [28] or the source code itself [13,14].

REFERENCES

1. Bennett, A.W., *Introduction to Computer Simulation,* West Publishing Co., New York, 1974.
2. Personal conversation with Dr. Eli Jacks, U.S. Weather Service, Silver Springs, Maryland.
3. McHenry, R.R., DeLeys, N.J., "Vehicle Dynamics in Single Vehicle Accidents - Validation and Extensions of a Computer Simulation," CAL Report No. VJ-2251-V-3, Cornell Aeronautical Lab, Buffalo, NY, 1968.
4. Segel, D.J., "Highway-Vehicle-Object Simulation Model," FHWA-RD-76-162, Federal Highway Administration, Washington, DC, 1976.
5. McHenry, R.R., "Development of a Computer Program to Aid the Investigation of Highway Accidents," CAL Report No. VJ-2979-V-1, Cornell Aeronautical Lab, Buffalo, NY, 1971.
6. McHenry, R.R., Naab, K.N., "Computer Simulation of the Crash Victim - A Validation Study," *Proceedings of the Tenth Stapp Car Crash Conference*, 1966.
7. Moncarz, H.T., Bernard, J.E., Fancher, P.S., "A Simplified, Interactive Simulation for Predicting the Braking and Steering Response of Comercial Vehicles," Highway Safety Research Institute, Univ. of Michigan, Report No. UM-HSRI-PF-75-8, Ann Arbor, MI, 1975.
8. MacAdam, C.C., Fancher, P.S., G.T., Gillespie, T.D., "A Computerized Model for Simulating the Braking and Steering Dynamics of Trucks, Tractor-semitrailers, Doubles, and Triples Combinations," Highway Safety Research Institute, University of Michigan, Report No. UM-HSRI-80-58, 1980.
9. Bowman,B.M., Bennett, R.O., Robbins, D.H., "MVMA Two Dimensional Crash Victim Simulation, Version 4," Highway Safety Research Institute, University of Michigan, Report No. UM-HSRI-79-5-2, Ann Arbor, MI, 1979.

10. *EDSVS Program Manual*, Engineering Dynamics Corporation, Lake Oswego, OR, 1985 (revised 1988).

11. *EDVTS Program Manual*, Engineering Dynamics Corporation, Lake Oswego, OR, 1985 (revised 1988).

12. *EDSMAC Program Manual*, Engineering Dynamics Corporation, Lake Oswego, OR, 1985 (revised 1989).

13. *EDSVS Source Code Listing*, Engineering Dynamics Corporation, Lake Oswego, OR, 1989.

14. *EDVTS Source Code Listing*, Engineering Dynamics Corporation, Lake Oswego, OR, 1989.

15. *EDSMAC Source Code Listing*, Engineering Dynamics Corporation, Lake Oswego, OR, 1989.

16. Day, T.D., Hargens, R.L., "An Overview of the Way EDSMAC Computes Delta-V," SAE Paper No. 880069, Society of Automotive Engineers, Warrendale, PA, 1988.

17. McHenry, R.R., Jones, I.S., Lynch, J.P., "Mathenatical Reconstruction of Highway Accidents - Scene Measurements and Data Processing System," CAL Report No. ZQ-5341-V-2, Cornell Aeronautical Lab, Buffalo, NY, 1974.

18. *Vehicle Dynamics Terminology*, SAE Technical Report No. J670e, Society of Automotive Engineers, Warrendale, PA, 1976.

19. Clark, S.D. (editor), *Mechanics of Pneumatic Tires*, US Department of Transportation, DOT HS 805 952, US Government Printing Office, Washington, DC, 1981.

20. Fiala, E., "Lateral Forces at the Rolling Pneumatic Tire," (transcribed for Ford Motor Co., Ref. No. 8753), Vienna, 1954.

21. *Mechanics of Heavy-duty Trucks and Truck Combinations*, Engineering Summer Conference, University of Michigan, June 22-26, 1981.

22. Runge, C., "Uber die numerische Auflosung von Differentialgleichungen," *Math. Ann.*, vol 46, 1895, pp. 167-178.

23. Kutta, W., "Beitrag zur naherungsweisen Integration totaler Differentialgleichungen," *Z. Math. Phys.* vol. 46, 1901, pp. 435-453.

24. Romanelli, M.J., "Runge-Kutta Methods for the Solution of Ordinary Differential Equations," *Mathematical Methods for Digital Computers*, Part 3, Wiley, New York, 1960, Chapter 8.

25. James, M.L., *Applied Numerical Methods for Digital Computation*, Harper and Row, New York, 1985, Chapter 6.

26. Ralston, A. "Numerical Integration Methods for the Solution of Ordinary Differential Equations," *Mathematical Methods for Digital Computers*, Part 3, Wiley, New York, 1960, Chapter 9.

27. Fryer, W., CALSPAN Corporation, adapted from the E.K. Blum modification of the Runge-Kutta fourth-order method found in *Mathematics of Computation*, April, 1962.

28. Hamming, R.W., *Numerical Methods for Scientists and Engineers*, Chapters 14-16, McGraw-Hill, New York, 1962.

INDEX

J

L

M

N

P

R

S

T

U

V

W

Y