The Engineering Society
For Advancing Mobility
Land Sea Air and Space®
400 COMMONWEALTH DRIVE, WARRENDALE, PA 15096-0001 U.S.A.

SAE Technical Paper Series

880069

An Overview of the Way EDSMAC Computes Delta-V

Terry D. Day and Randall L. Hargens
Engineering Dynamics Corp.

**Reprinted from SP-733—
Accident Reconstruction — State-Of-The-Art**

**International Congress and Exposition
Detroit, Michigan
February 29 — March 4, 1988**

The papers included in this volume are abstracted and indexed in the SAE Global Mobility Database

SAE GLOBAL MOBILITY DATABASE

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

ISSN 0148-7191
Copyright 1988 Society of Automotive Engineers, Inc.

Positions and opinions advanced in this paper are those of the author(s) and not necessarily those of SAE. The author is solely responsible for the content of the paper. A process is available by which discussions will be printed with the paper if it is published in SAE Transactions. For permission to publish this paper in full or in part, contact the SAE Publications Division.

Persons wishing to submit papers to be considered for presentation or publication through SAE should send the manuscript or a 300 word abstract of a proposed manuscript to: Secretary, Engineering Activity Board, SAE.

Printed in U.S.A.

880069

An Overview of the Way EDSMAC Computes Delta-V

Terry D. Day and Randall L. Hargens
Engineering Dynamics Corp.

ABSTRACT

The EDSMAC personal computer program for use by accident investigators is described. The input data requirements are reviewed. The general calculation procedures are discussed and the specific procedures for computing delta-V are explained in detail. The method, based on equalizing the force between the vehicles at all times during the impact phase, is seen to be simple in concept but extremely complex in practice. The numerical and graphical output and warning messages are reviewed. Applications of the program are illustrated. The major benefit of EDSMAC is the ability, using graphics, to provide an analytical method illustrating how an accident may, or may not, have occurred.

DELTA-V IS DEFINED as the change in the velocity of a vehicle's occupant compartment during the collision phase of a motor vehicle crash (i.e., from the moment of initial contact until the moment of separation). The delta-V is frequently used as an indicator of the severity of impact because it approximates the speed of the *second collision* - the collision between the occupant and vehicle interior - that causes injury [1,2,3]*. If the pre-impact speeds and exterior properties of two colliding vehicles are known or can be estimated, the EDSMAC computer program [4] can estimate the delta-V for both vehicles. The program can also be used to produce a time-motion history of the entire accident sequence.

The purpose of this paper is to describe the calculation procedures used by EDSMAC, and particularly those used to compute the delta-V.

*Numbers in brackets designate references at the end of the paper.

PROGRAM OVERVIEW

EDSMAC is an IBM-compatible personal computer program for use by accident investigators. It was developed from the SMAC (Simulation Model of Automobile Collisions) mainframe computer program developed under contract to NHTSA by Calspan [5] as a tool to help improve highway safety. Except where noted, all EDSMAC calculations are identical to those used in SMAC. However, the input and output have been substantially enhanced to take advantage of the personal computer environment.

EDSMAC is a simulation program - that is, the investigator supplies the conditions believed to precede the crash. EDSMAC then uses vehicle properties, physics and numerical integration to create the vehicle paths and damage profiles which would be expected to result from those initial conditions. It is important to observe that EDSMAC, as a simulation, analyzes a crash in precisely the opposite way when compared to a reconstruction program, such as EDCRASH [6], which estimates the initial conditions. This concept is illustrated by the fact that EDCRASH creates a complete EDSMAC input file.

Program Input

Input data is supplied to the EDSMAC program during an interactive input session; that is, each input variable is checked as it is entered to insure the data is reasonable and has the correct syntax. The input data is divided into six data categories, described below.

Simulation Variables - Numerical integration results in the calculation of vehicle positions and velocities at discrete time intervals. Therefore, EDSMAC requires integration time intervals as input data for the various phases (collision, separation and trajectory) of the accident. In addition, the maximum length of the simulation and the time interval for output must be supplied.

The simulation should end when both vehicles come to rest. However, since the velocities are computed from accelerations at discrete time intervals, it is likely the simulated vehicles will never come to rest. Instead, they will continue to move back and forth at a slow speed indefinitely. To prevent this, small values for linear and angular velocity, called termination velocities, must be supplied as input. If the simulation velocities drop below these values, it is assumed the vehicle has stopped moving.

Initial positions and velocities - EDSMAC requires the initial X and Y coordinates and heading angle for each vehicle, as well as the forward, lateral and angular velocities. These data are entered according to the coordinate system convention adopted by SAE [7]. The initial position need not be the position at impact. Therefore, a complete time history of the pre-impact, impact, and post-impact events can be simulated.

Vehicle Data - The simulation requires the weight, yaw inertia, wheel base, track width, overall length, overall width, and CG location for each vehicle. Tire cornering stiffness must also be supplied. At the user's request, this data can be loaded automatically from a table of vehicle parameters according to wheelbase. Selected parameters can then be modified where the actual data is known.

Tire/road Data - Tire/road friction must be supplied for each vehicle. A terrain boundary, separating two zones on the highway having different friction properties, can also be supplied.

Crush Data - The method for computing the collision force assumes the vehicle has a structurally homogeneous exterior which is modelled by force vectors placed at specified intervals about the vehicle exterior. The end points of these force vectors specify the damage profile (the shape and location of damage). Thus, a crush stiffness coefficient, the spacing between the force vectors, the crush increment, and maximum force error between vehicles must be supplied. These variables are described in this paper (see **Computation of Delta-V**).

The collision model also allows for restitution, and requires three coefficients which are used to determine elastic rebound characteristics.

Braking and steering for both vehicles - Computation of the tire force requires the braking (or accelerating) and steering at each wheel be entered. This information is provided by the table entry of braking vs. time and steering vs. time. Unlike the original SMAC program, EDSMAC allows non-uniform time increments for brake and steering data and uses linear interpolation to compute the braking and steering levels at times between table entries.

EDSMAC has a separate Brake/Steer Table, allowing up to 15 separate braking and steering entries, for each vehicle.

GENERAL CALCULATION PROCEDURES

All computer programs incorporate an input phase (the portion of the program devoted to entering and organizing the user's input data), a processing phase (the portion which performs the desired calculations), and an output phase (the portion which displays the results in a desired format). The following discussion pertains specifically to the processing phase.

EDSMAC, like other simulation programs, uses the laws of physics to compute vehicle forces and their resulting effect on motion (i.e., vehicle accelerations). Numerical integration is then used to compute positions and velocities at specific time intervals. A time-motion history of the results can be printed or graphed at selected time intervals.

The EDSMAC program is modular: Certain routines in the program are devoted to certain calculation procedures. Unlike computer reconstruction programs such as EDCRASH, the calculation procedures are quite lengthy and extremely repetitive. However, the concepts are quite simple. An overview of the general calculation procedures which occur during the processing phase is provided by the flow chart (see figure 1, below).

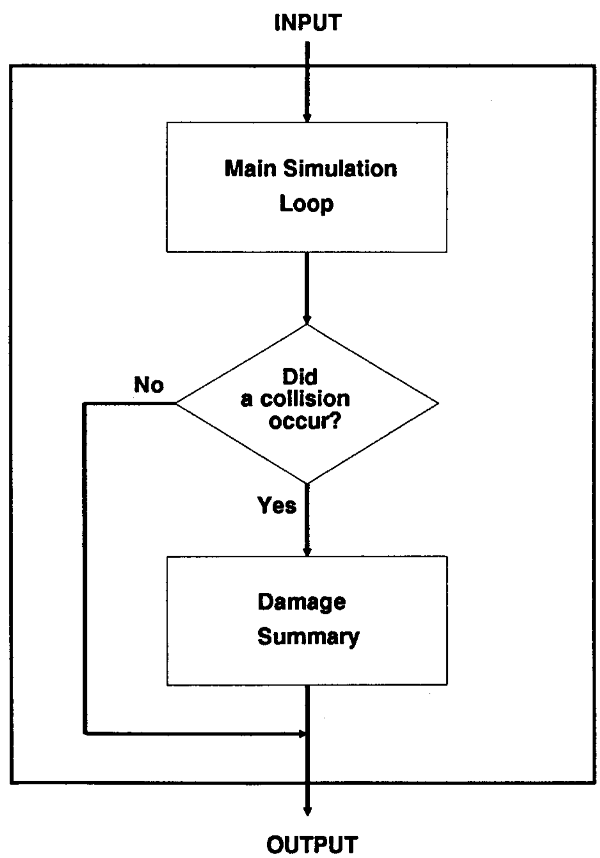


Figure 1 - EDSMAC Processing Phase Flow Chart

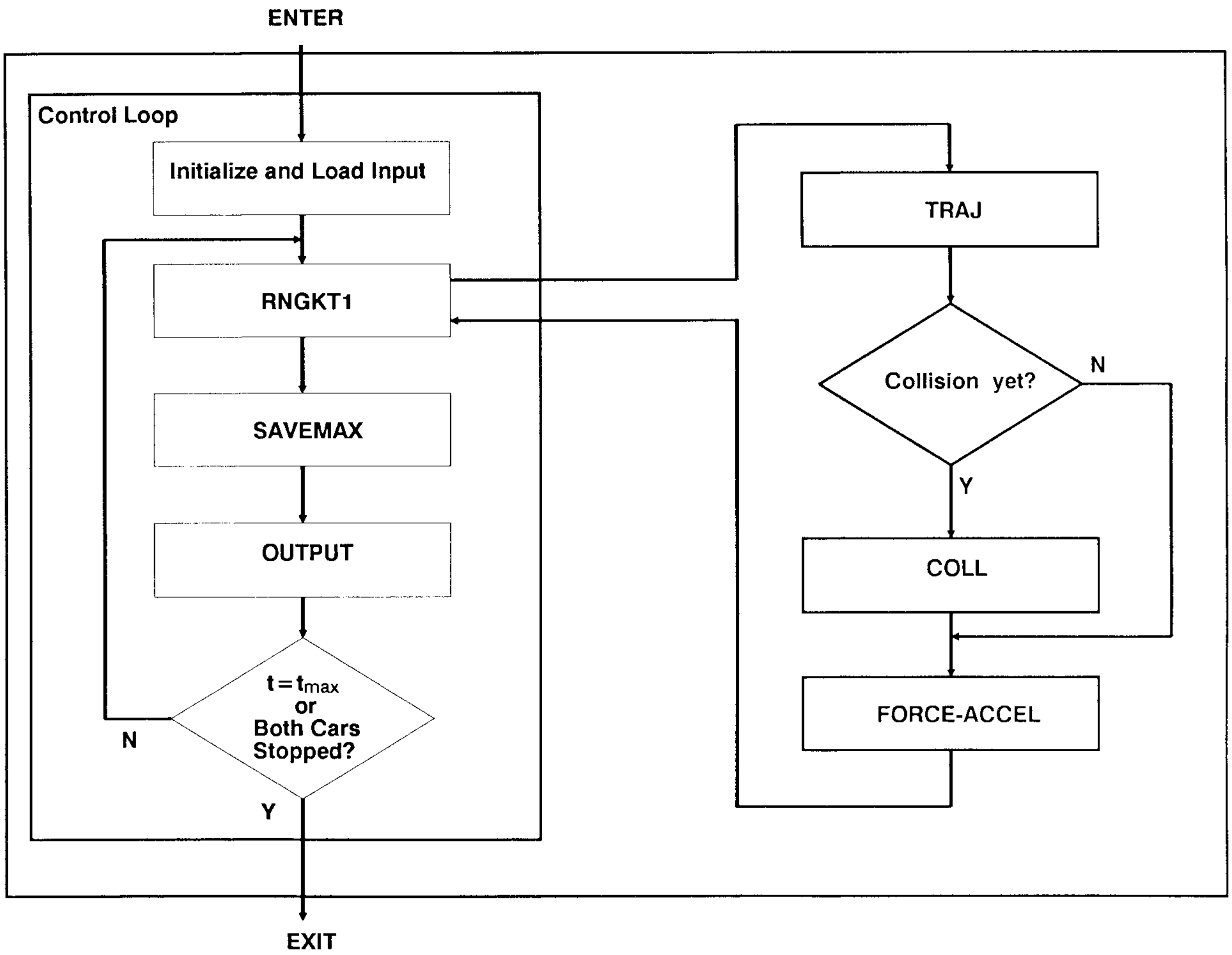


Figure 2 - Main Simulation Loop Flow Chart

As shown in figure 1, there are two major calculation procedures during the processing phase. These procedures are (1) the Main Simulation Loop and (2) the Damage Summary. These two procedures are described below.

Main Simulation Loop

Most of the work is done in the Main Simulation Loop. The Main Simulation Loop flow chart (see figure 2) shows there are actually four major procedures within this loop. These calculation steps are performed at each successive time increments until both vehicles stop moving or until the maximum simulation time is reached, which ever occurs first. Each of these procedures is described below.

Control Loop - This loop first initializes all variables and loads the input data file. Then, the calculations begin.

The numerical integration process provides the step-by-step control over program execution. EDSMAC uses the Runge-Kutta, fourth-order numerical integration routine called RNGKT1 [8]. RNGKT1 increments the simulation time by the specified time step. Then RNGKT1 passes control to other routines which compute tire and collision forces and resulting vehicle accelerations. RNGKT1 then uses numerical integration to determine the vehicle velocities and positions to be used during the next time step.

A routine, called SAVEMAX, evaluates the motion which occurred during the current time step and compares these results with what happened previously. In particular, SAVEMAX saves the magnitude and direction of the individual impulse accelerations. As will be shown later, the delta-V is calculated from these accelerations.

The final routine in the control loop is called OUTPUT. This routine is executed whenever the current simulation time has reached the specified print time interval and causes the vehicle positions, velocities and accelerations to be written to the output file.

If the simulation time has reached the maximum length specified by the user, or if both vehicles have stopped moving (as specified by the termination velocities), the Main Simulation Loop calculations stop and control is passed to the Damage Summary.

Tire Force Computation - The Fiala [9] tire model and a friction circle (the vector sum of forward and lateral components of tire force may not exceed the available friction force) are used to compute the force at each tire from the current vertical tire load, friction, slip angle and cornering stiffness. This model is part of a routine, called TRAJ, which computes the forces at the individual tires during the current time interval.

Collision Force Computation - This routine, called COLL, computes the inter-vehicle force due to the collision for the current time interval (see **Computation of Delta-V**).

Acceleration Computation - This routine, called FORCE-ACCEL, adds the individual (tire and collision) forces and uses Newton's 2nd law ($\Sigma F = ma$) to compute the vehicle linear and angular accelerations resulting from the forces and moments acting on the vehicles.

Damage Summary

The Damage Summary occurs after the conclusion of the Main Simulation Loop. The Damage Summary flow chart (see figure 3) shows there are only two procedures within this loop: computing the damage ranges and determining the associated Collision Deformation Classification (CDC), a 7-character code describing the vehicle damage [10]. The damage ranges and CDC routines are performed once for each vehicle.

Damage Ranges - By the end of the Main Simulation Loop, a vehicle damage profile has been determined from the final lengths of the collision force vectors. This routine, called RANGDAM, analyzes and organizes the vector end points to determine the number of individual damage locations, or *ranges*, on the vehicle exterior.

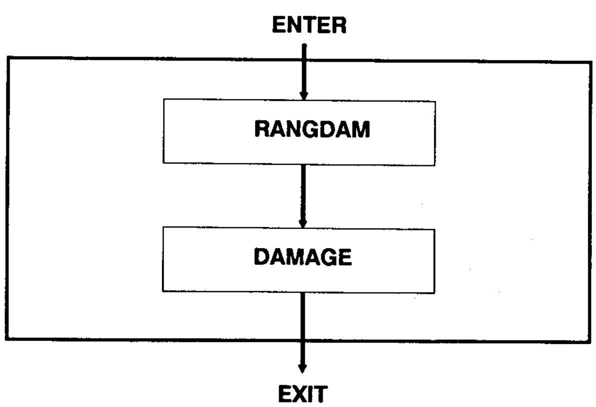


Figure 3 - Damage Summary Flow Chart

CDC - Each individual damage range is compared with the original vehicle dimensions to compute a CDC. The routine, called DAMAGE, which accomplishes this task is a logical routine which compares each damage range with the damage limits identified by specific CDC codes.

A vehicle may have more than one damage range and CDC. For example, separate damage ranges can occur during secondary impacts. A good example of this occurs during intersection collisions when the resulting vehicle rotation (clockwise for one vehicle and counter-clockwise for the other) causes the rear of the vehicles to slap together. Additional CDCs are computed for each additional damage range.

COMPUTATION OF DELTA-V

The basic data for the delta-V computation are produced by the COLL routine. COLL computes the inter-vehicle force at discrete time intervals specified by the numerical integration process. The flow chart in figure 4 describes this process. The main result of this process is the computation of inter-vehicle force for the current time interval.

Computing the Inter-vehicle Force

This section describes how the inter-vehicle force is computed for each point in time. The process takes many steps. The majority of the task is spent simply in keeping track of various angles and collision vectors.

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 5). 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 5, 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.

Calculating the Radial Vectors

A set of equally spaced radial vectors is next established (see figure 6). 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

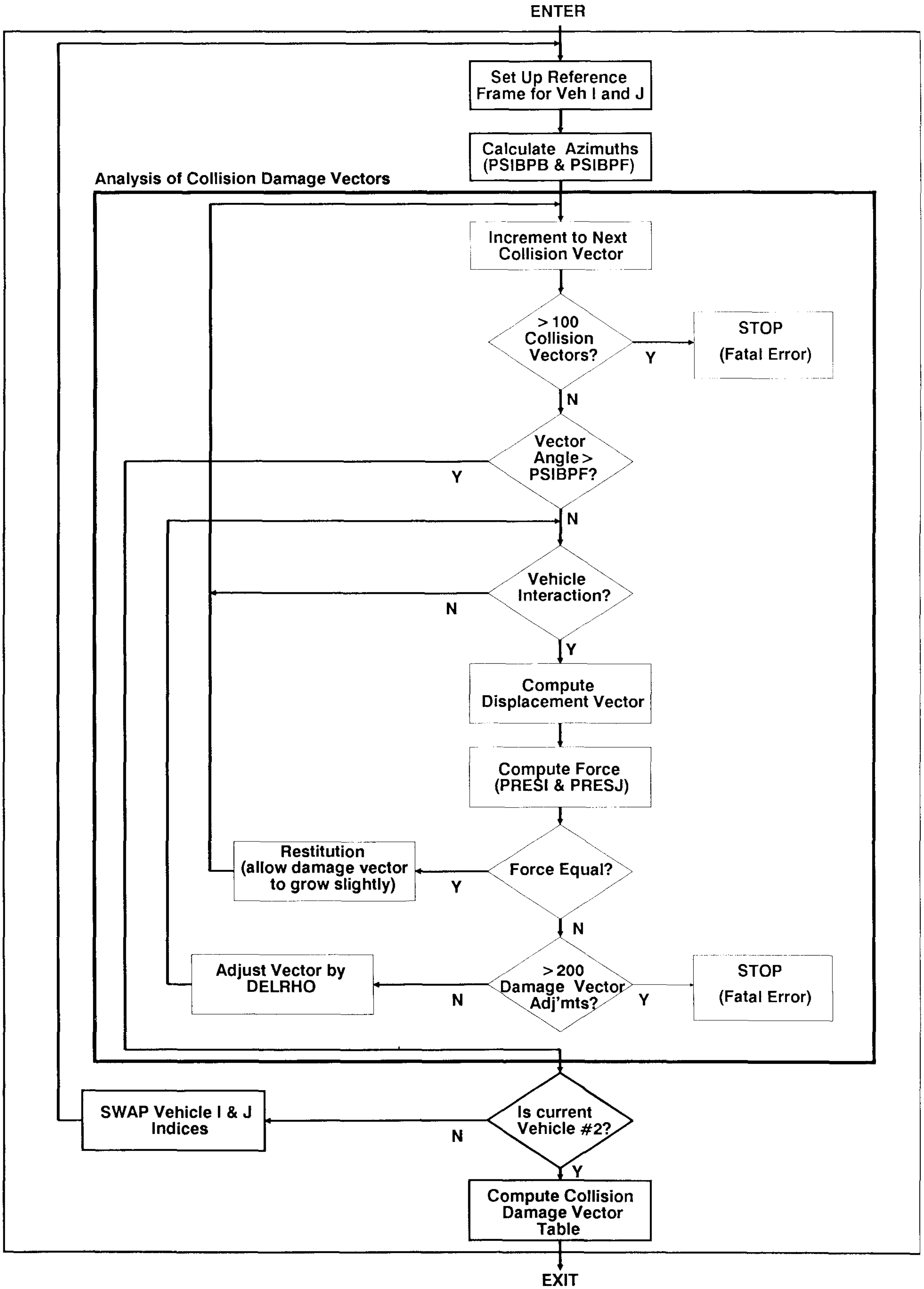


Figure 4 - COLL Routine for Inter-vehicle Force Computation

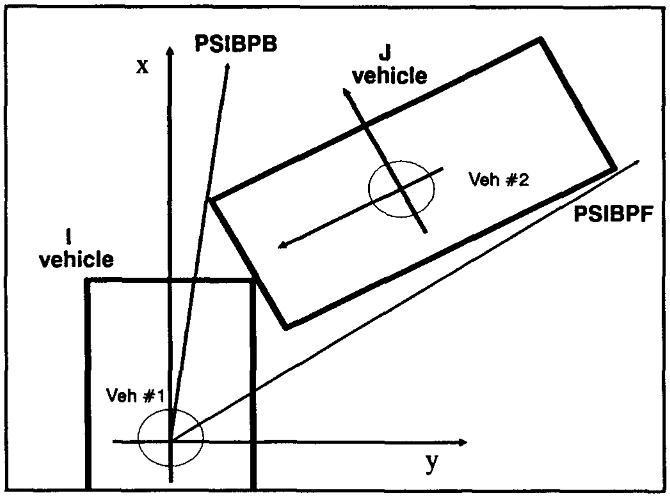


Figure 5 - Viewing vehicle no. 2 from vehicle no. 1's reference frame. Note azimuth angles, PSIBPB & PSIBPF, which define the maximum possible range of damage.

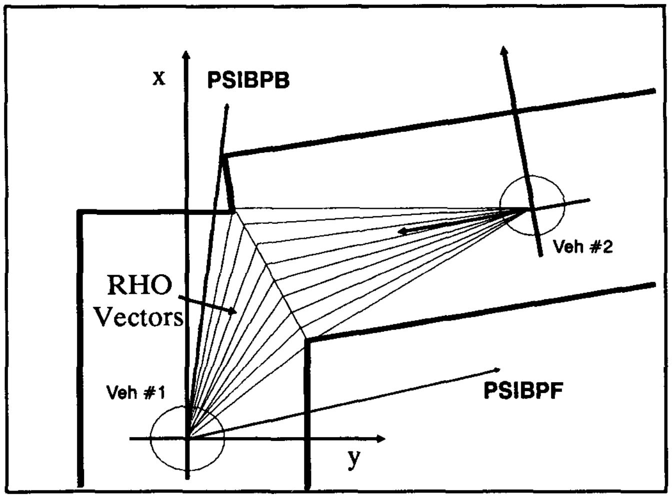


Figure 6 - RHO vectors from base vehicle within damage range defined by PSIBPB and PSIBPF. Vectors are equally spaced DELPSI degrees.

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 7, it 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 8). 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

$$\text{PRESI} = \text{AKVI} * \text{DELTAI} \qquad \text{(base vehicle)}$$

and

$$\text{PRESJ} = \text{AKVJ} * \text{DELTAJ} \qquad \text{(other vehicle)}$$

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 8, 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 equal-

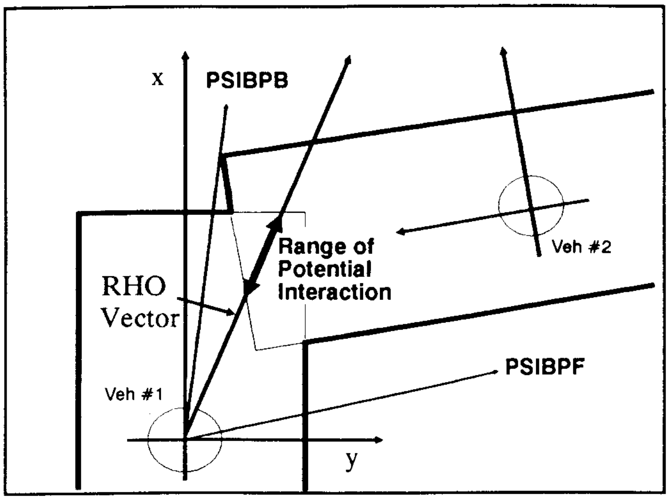


Figure 7 - A collision is confirmed by seeking interaction along a particular RHO vector, i.e., a point on the vector lies within the perimeter of both vehicles.

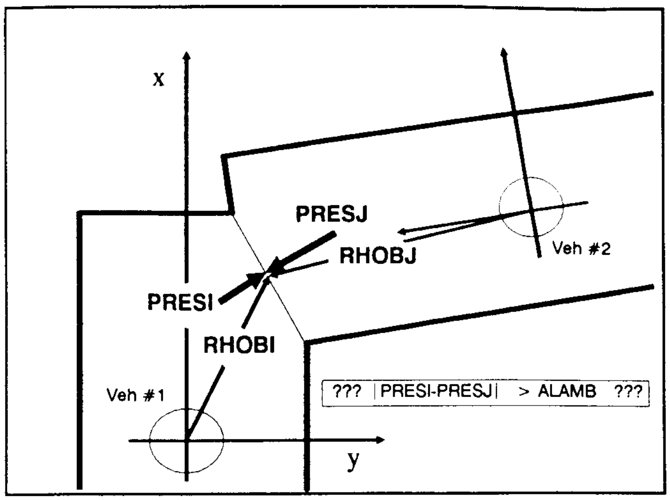


Figure 8 - Computing the displacement vectors. The displacement from the original surface is used to compute the collision force at each interaction point for both vehicles.

ize PRESI and PRESJ. If more than 200 adjustments occur, a fatal error is issued and calculations stop.

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.

Restitution

Elastic restitution is modelled by allowing the length of the RHO vector to increase.

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 9.

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 10) 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

Computing the Tire Force

The Fiala tire model is used to compute the force at each tire from the current tire load, friction, slip angle and cornering stiffness. The sum of the forces at the individual tires is used to compute the total tire force and moment acting on the vehicle CG (see figure 11). During a given integration time step, these forces and moments are:

SFXT - sum of F_x from tires

SFYT - sum of F_y from tires

SFNT - sum of moments about z-axis

Computing the Total Force

The total force on the vehicle, neglecting aerodynamic drag, is the sum of the collision and tire forces, or (see next page):

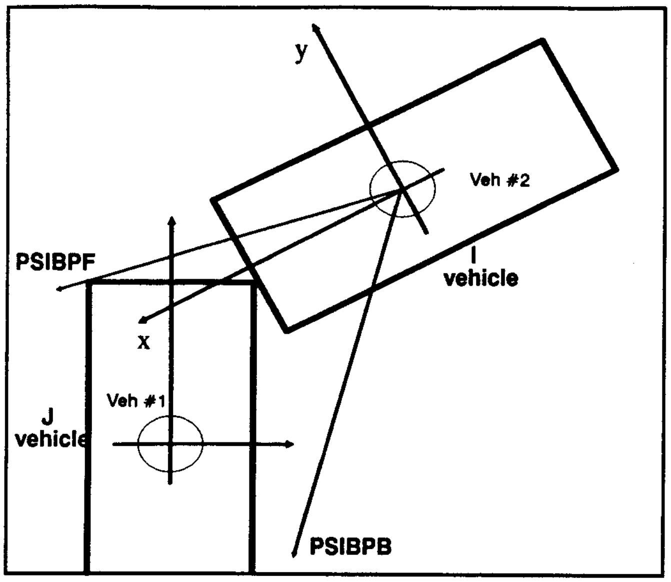


Figure 9 - After calculating from vehicle no. 1's reference frame, the vehicle indices, I and J, are swapped and calculations are repeated from vehicle no. 2's reference frame.

(Total force on vehicle, cont.):

$$SFX = SFXC + SFXT$$

$$SFY = SFYC + SFYT$$

$$SFN = SFNC + SFNT$$

Computing the Acceleration

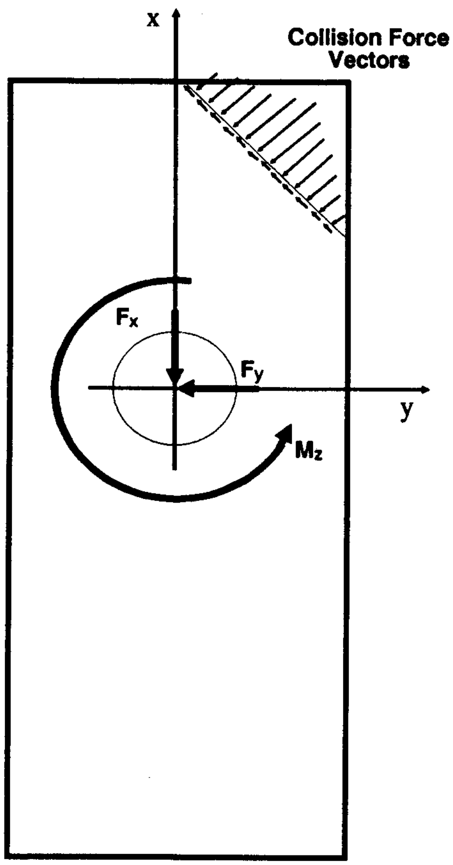
The total force on each vehicle is used to compute its forward, lateral and angular acceleration according to Newton's 2nd law:

$$ACC_{angular} = SFN/I_z$$

$$ACC_{Fwd} = SFX/m + V\text{-vel}*PSID$$

$$ACC_{Lat} = SFY/m - U\text{-vel}*PSID$$

where I_z is the vehicle's yaw inertia, m is the mass, U- and V-vel are the forward and lateral velocities, respectively, and PSID is the angular velocity.



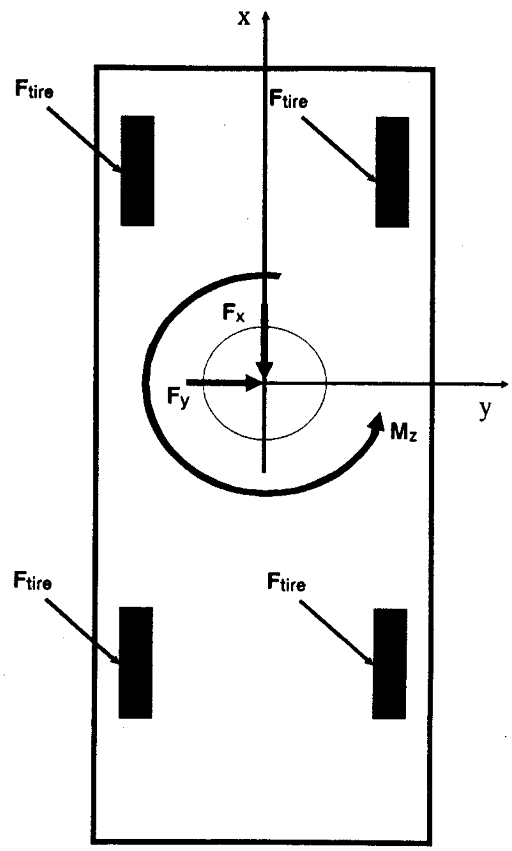


Figure 10 - Sum of forces and moments resulting from collision forces

Figure 11 - Sum of forces and moments resulting from tire forces

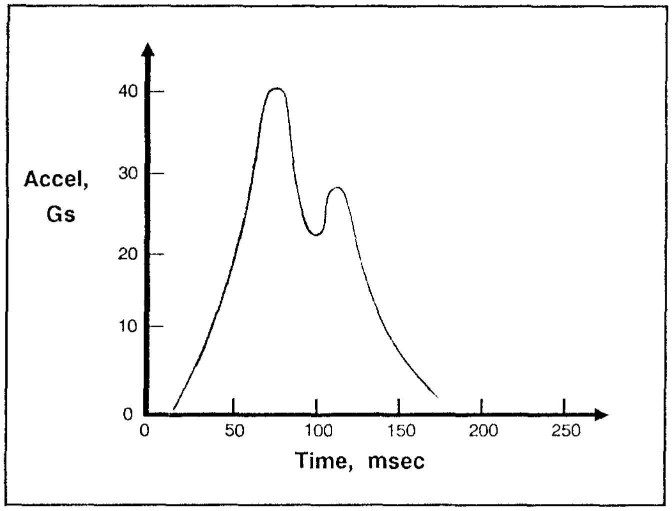


Figure 12 - Typical acceleration vs. time history for duration of impact.

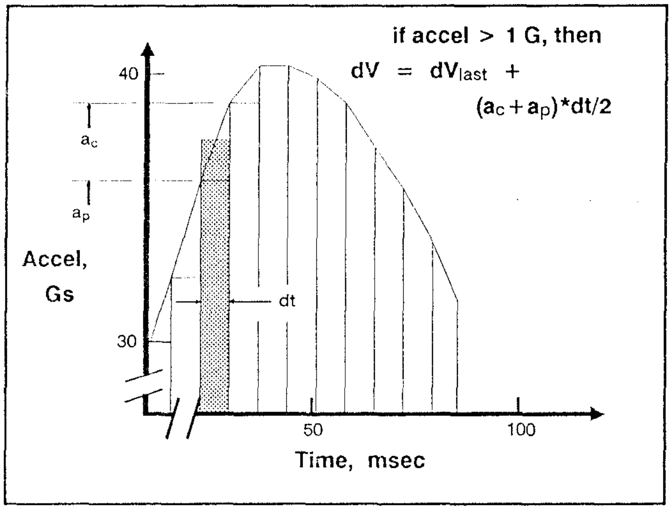


Figure 13 - The delta-V is computed from the area under the acceleration vs. time curve. This process occurs during the SAVEMAX routine.

These accelerations are computed for each collision time step, DTCOLL. The result is an acceleration vs. time history (see figure 12). The numerical integration routine, RNGKT1, then computes velocity vs. time and position vs. time histories from these accelerations.

Delta-V

From basic physics, it is known that $dV = \int a dt$. Graphically, this is simply the area under the acceleration vs. time curve (figure 12). At the conclusion of each integration time step, control is returned to the Main Loop. During the collision phase, the incremental areas for each time step are summed by using the average acceleration for the current time increment (figure 13). The delta-V is equal to the total area under the curve at the conclusion of the collsion phase.

The X and Y components of the acceleration vs. time history are also used to keep track of the direction of the delta-V. From this direction, the clock direction of the Principal Direction of Force (PDOF) is computed. If the collision results in more than one CDC, a separate delta-V is determined for each individual CDC only if the individual CDCs have different PDOF clock directions.

MODIFICATIONS

Five modifications were made to the original SMAC mainframe numerical calculations. These changes were:

- Revised braking and steering algorithm
- Revised terrain boundary algorithm
- Correction to COLL subroutine
- Correction to DAMAGE subroutine
- Correction to RANGDAM subroutine

Braking and Steering Modification

The SMAC braking and steering algorithm used quadratic interpolation to determine the level of input between table values. This method required constant time increments between table values, and had some other features making it difficult to use.

EDSMAC uses linear interpolation between table values, allowing the time increment to be varied. This method allows for a more flexible method of table entry. A validation study comparing the two methods was performed for all RICSAC [11] staged collisions. No difference associated with the linear interpolation method was observed in the results.

Terrain Boundary Modification

The original SMAC program used a terrain boundary method incorporating a *mirror image* by creating a second terrain boundary reflected about the earth-fixed X axis. This method was intended for the simulation of highway medians. Unfortunately, the method was not properly explained in the NHTSA documentation [4] and has been improperly applied by accident investigators using SMAC and CRASH3. The associated results were frequently based on the wrong tire/road friction coefficient.*

EDSMAC removed the mirror image, eliminating the potential for error described above. However, a vehicle returning to original the friction surface after crossing a median cannot be simulated. This limitation was justified by the small number of highways having a flat highway-median-highway cross-section required by the TRAJ routine which computes the tire forces acting on the vehicle.

*The original method, describing the mirror image terrain boundary, is described in a footnote found in reference 12.

Time (Sec)	X #1 (ft)	Y #1 (ft)	PSI #1 (deg)	X #2 (ft)	Y #2 (ft)	PSI #2 (deg)
0.000	0.00	31.00	0.00	-40.00	30.00	0.00
0.010	0.00	31.00	0.00	-39.49	30.00	0.00
0.020	0.00	31.00	0.00	-38.97	30.00	0.00
0.030	0.00	31.00	0.00	-38.46	30.00	0.01
0.040	0.00	31.00	0.00	-37.95	30.00	0.03
0.050	0.00	31.00	0.00	-37.43	30.01	0.06
0.060	0.00	31.00	0.00	-36.92	30.01	0.09
0.070	0.00	31.00	0.00	-36.41	30.01	0.14
0.080	0.00	31.00	0.00	-35.90	30.02	0.19
0.090	0.00	31.00	0.00	-35.38	30.03	0.25
0.100	0.00	31.00	0.00	-34.87	30.03	0.32
0.110	0.00	31.00	0.00	-34.36	30.04	0.40
0.120	0.00	31.00	0.00	-33.85	30.05	0.49
0.130	0.00	31.00	0.00	-33.34	30.06	0.58
0.140	0.00	31.00	0.00	-32.82	30.07	0.69
0.150	0.00	31.00	0.00	-32.31	30.08	0.80
0.160	0.00	31.00	0.00	-31.80	30.10	0.92
0.170	0.00	31.00	0.00	-31.29	30.11	1.04
0.180	0.00	31.00	0.00	-30.78	30.13	1.18
0.190	0.00	31.00	0.00	-30.27	30.14	1.32
0.200	0.00	31.00	0.00	-29.76	30.16	1.47
0.210	0.00	31.00	0.00	-29.25	30.18	1.62
0.220	0.00	31.00	0.00	-28.74	30.20	1.79
0.230	0.00	31.00	0.00	-28.23	30.22	1.95
0.240	0.00	31.00	0.00	-27.73	30.25	2.13
0.250	0.00	31.00	0.00	-27.22	30.27	2.31
0.260	0.00	31.00	0.00	-26.71	30.29	2.49
0.270	0.00	31.00	0.00	-26.20	30.32	2.69
0.280	0.00	31.00	0.00	-25.70	30.35	2.88
0.290	0.00	31.00	0.00	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	-74.02	9.29	36.41	13.45
1.840	33.41	25.87	-74.85	9.38	36.43	13.45
1.850	33.55	25.79	-75.68	9.47	36.45	13.45
1.860	33.70	25.71	-76.52	9.55	36.47	13.45
1.870	33.84	25.64	-77.35	9.64	36.49	13.46
1.880	33.98	25.56	-78.19	9.72	36.51	13.46
1.890	34.11	25.48	-79.03	9.81	36.53	13.46
1.900	34.25	25.40	-79.87	9.89	36.55	13.46
1.910	34.38	25.33	-80.71	9.97	36.57	13.46
1.960	35.01	24.94	-84.94	10.36	36.67	13.47
2.010	35.60	24.57	-89.18	10.71	36.75	13.48
2.060	36.13	24.20	-93.44	11.03	36.83	13.49
2.110	36.61	23.85	-97.70	11.31	36.90	13.49
2.160	37.04	23.51	-101.97	11.57	36.96	13.50
2.210	37.41	23.19	-106.24	11.79	37.01	13.50
2.260	37.74	22.89	-110.50	11.97	37.06	13.50
2.310	38.03	22.62	-114.76	12.13	37.09	13.50
2.360	38.26	22.37	-119.01	12.25	37.12	13.50
2.410	38.47	22.15	-123.16	12.33	37.15	13.48
2.460	38.65	21.96	-127.05	12.39	37.16	13.50
2.510	38.81	21.78	-130.64	12.41	37.16	13.51
2.560	38.97	21.63	-133.91	12.41	37.16	13.51
2.610	39.10	21.49	-136.86	12.41	37.16	13.51
2.660	39.22	21.37	-139.42	12.41	37.16	13.51
2.710	39.33	21.25	-141.66	12.42	37.15	13.51
2.760	39.41	21.16	-143.57	12.42	37.15	13.51
2.810	39.47	21.08	-145.17	12.42	37.15	13.51
2.860	39.51	21.02	-146.45	12.42	37.15	13.51
2.910	39.54	20.97	-147.39	12.42	37.15	13.51
2.960	39.56	20.95	-147.99	12.42	37.14	13.51
3.010	39.56	20.94	-148.16	12.42	37.14	13.51

Figure 15 - Variable output from EDSMAC. The table displays all simulation variables as a function of time. The *spreadsheet-style* format allows columns of data to be rearranged and displayed at desired time intervals.

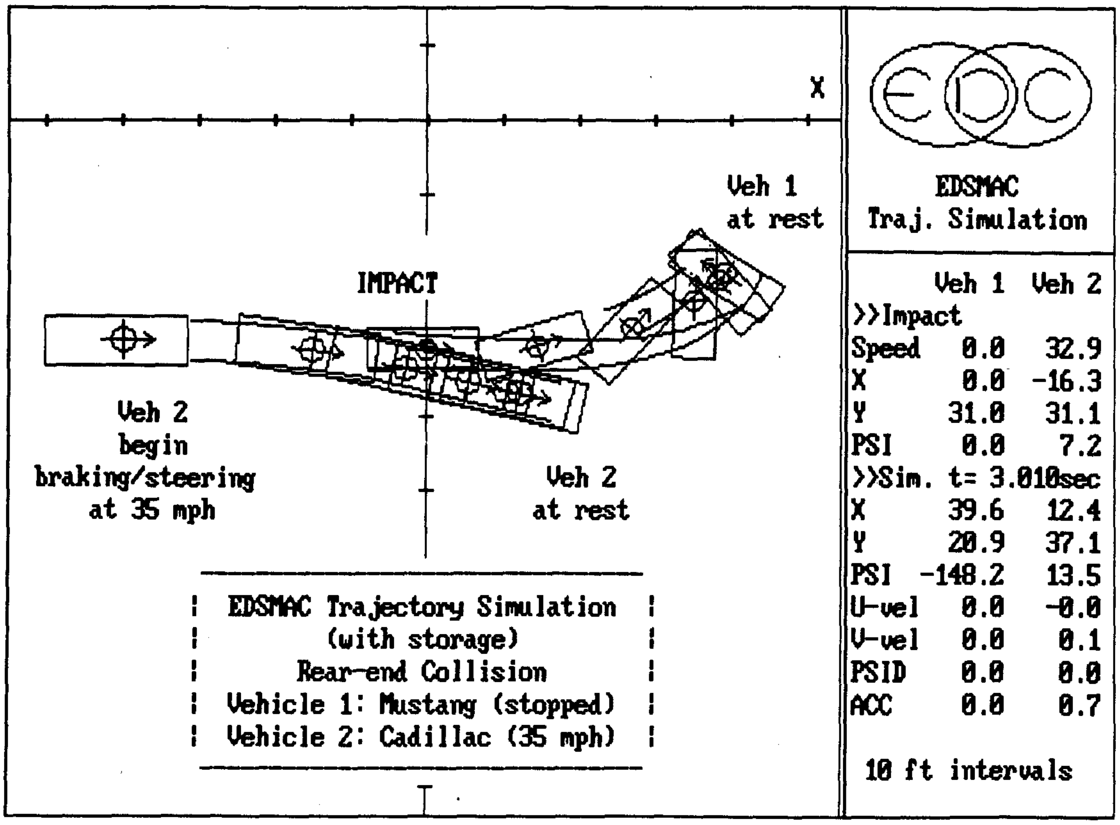


Figure 16 - Trajectory Simulation

an error message is issued (see **Warning Messages**). This message is falsely issued if the damage midpoint is slightly greater than 0 (say, 10 degrees) and the CDC clock direction is 11-o'clock. Note that the true difference is only (10-(-30)) = 40 degrees. However, since the angle is computed from the clock direction (11-o'clock), the result is 330 degrees (30 degrees/hour times 11 hours) and the difference between the PDOF and PSIM is incorrectly computed as (10-(330) = -320 degrees. Since the absolute value of -320 degrees is greater than the allowable angle of 60 degrees, the error message is falsely issued.

The problem occurs just before FORTRAN statement number 54 in subroutine RNGDAM:

```
TEST = PSIMT - PSTRY
```

where PSIMT is the temporary damage midpoint and PSTRY is the angle of the PDOF.

To solve the problem, the following line should be inserted directly after the above statement:

```
IF (TEST .GE. PI) THEN
    TEST=TEST - TWOPI
```

where TWOPI = 2*PI. This statement changes the value of TEST from 330 degrees to -30 degrees, which is only 40 degrees from PSIM.

OUTPUT

The results of analysis are displayed in three forms: numerical, graphical and warning messages. Each form of output serves a different purpose, which is described below.

Numerical Output

EDSMAC output can be routed to the screen or printer and is formatted accordingly. A complete or abbreviated listing may be selected. The complete listing includes an echo of all input data, the accident history (figure 14), the summary of damage ranges (which includes the CDC and delta-V) and *Variable Output*. The abbreviated listing displays only the accident history. Numerical output always begins with a display of all warning messages (described later).

Variable output is a custom spreadsheet used to display the position, velocity and acceleration vs. time information in a column format. These columns can be arranged in any order for viewing on the screen or for printing. Since the total amount of data can frequently fill a 50-page book, Variable Output is a convenient way to organize and view only the desired data. Figure 15 shows an example of results printed from Variable Output.

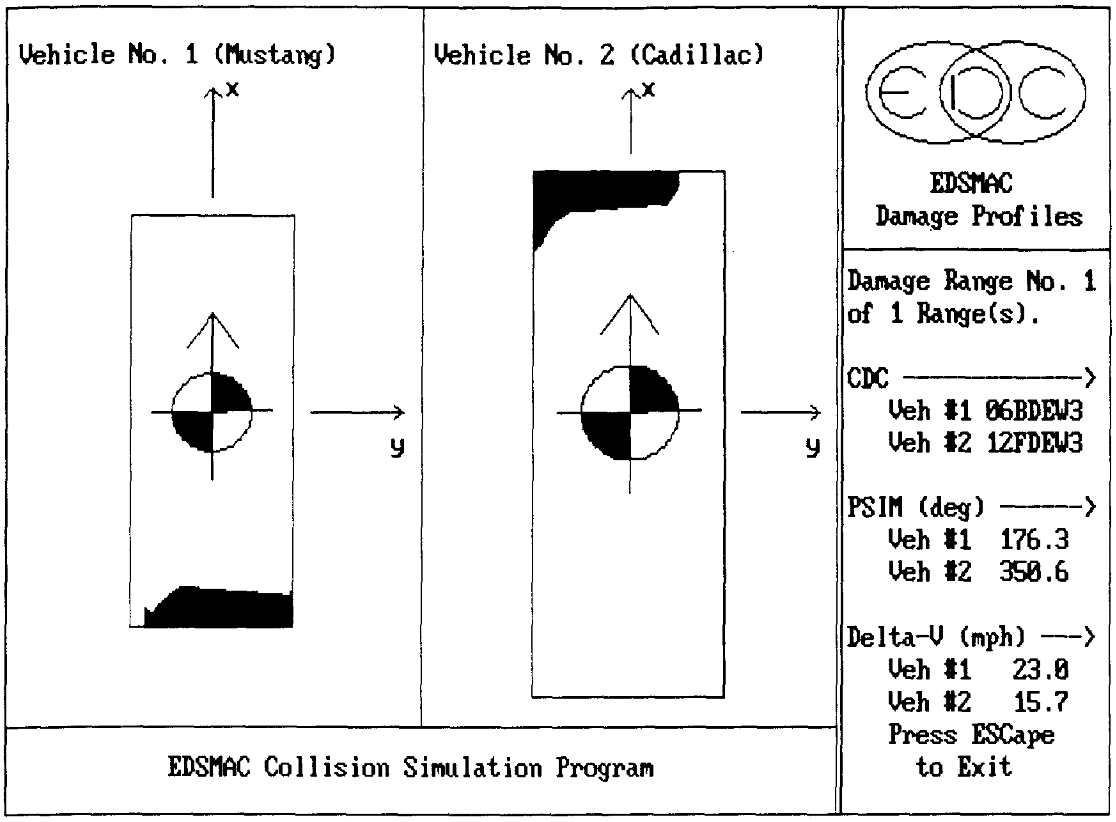


Figure 17 - EDSMAC Damage Profiles

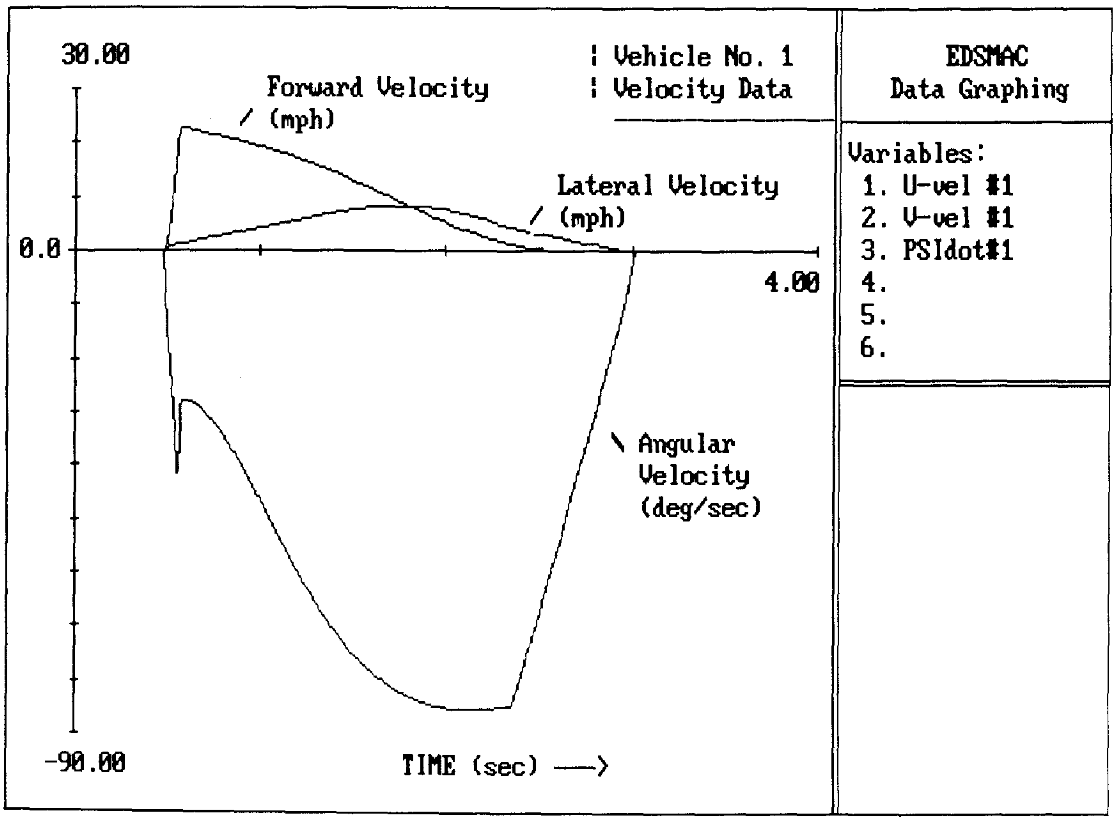


Figure 18 - EDSMAC Data Graphing

Graphical Output

Three types of graphics are available: trajectory simulations (with storage or animation), damage profiles, and a data graphing utility. Each is described below.

Trajectory simulation - A trajectory simulation uses the position (X,Y, and heading angle) vs. time history and vehicle exterior dimensions to show the vehicle moving from the start until the end of the simulation (see figure 16). This is an extremely useful and convenient way of using the simulation results. The time increment for displaying the vehicles can be set and the user can decide whether or not to display simulated skid marks left by the tires as they exceed the available friction.

EDSMAC allows two forms of trajectory simulations. The *With Storage* option displays the vehicle at each selected time interval, resulting in a series of vehicles being displayed over the path. The *Without Storage* option momentarily displays both vehicles, then erases them before displaying the vehicles at their new locations. This method results in a computer animation of the accident which can be displayed and saved on a VCR.

The purpose of the trajectory simulation is to confirm the simulated vehicle paths match the actual vehicle paths. If not, adjustments must be made and another run performed. The graphical trajectory simulation makes confirmation and adjustments much easier when compared to the use of numerical path data alone.

Damage Profiles - The damage profile computed from the individual collision vectors is used to illustrate the vehicle damage. This is accomplished by drawing a line connecting the ends of all the damage vectors and shading the resulting area. The CDC and delta-V are also displayed (see figure 17).

The purpose of this display is to confirm the simulated vehicle damage looks like the actual vehicle damage. If it does not, adjustments must be made and another run must be performed. Again, confirmation and adjustments are much easier when compared to the use of the numerical damage data alone.

Data Graphing - This utility program can be used to graph any of the simulation vs. time variables (position, velocity, acceleration, sideslip and other data).

Up to six variables can be graphed at once. The data are automatically scaled. Figure 18 shows a graph of forward, lateral, and angular velocities using the EDSMAC Data Graphing utility.

Warning Messages

EDSMAC contains all the error messages which existed in the original SMAC code. Some of these messages are fatal, and cause the calculations to stop before the end of the run. However, the vast majority are programming diagnostics. Each of the warning messages possibly occurring during the calculation of delta-V is described in this section. Use figures 5 through 9, which illustrate the collision model, to assist in this discussion.

Message: *An error occurred during the collision phase at t = *.** seconds because RHOBIT became negative.* (Fatal)

RHOBIT is the maximum possible length of a RHO vector (i.e., the length of a line from the CG to the original (pre-collision) vehicle exterior). This message occurs when the exterior body dimensions (CG to front, CG to rear and/or overall width) are zero or negative. Since the EDSMAC input session checks the values first, this message can no longer occur.

Message: *An error occurred during the collision phase at t = *.** seconds because the longitudinal and lateral components of damage were equal to zero.* (Fatal)

PSIB is the angle of a specified collision vector, RHO. IPSIB is the vector number (remember, up to 100 RHO vectors are allowed - see **Computing the Displacement Vector**). The sine and cosine of the angle, PSIB, are placed in temporary arrays, TSPSIB(IPSIB,I) and TCPSIB(IPSIB,I), respectively. The longitudinal and lateral components of the RHO vector can both be zero only if the sine and cosine of PSIB are simultaneously zero. Since this is not possible, the only way this error can occur is if neither TSPSIB() nor TCPSIB() were properly set after being initialized. Should this occur, a logic error in the routine is indicated.

Message: *An error occurred during the collision phase at t = *.** seconds because the damage increment was too coarse.* (Fatal)

The angle between RHO vectors for the base vehicle is set by the user (DELPSI). The associated angle for the other vehicle is dependent upon the collision geometry. If the resulting angle is too large, it may not be possible to equalize PRESI and PRESJ. This error message is displayed when this condition occurs.

Message: *An error occurred during the collision phase at t = *.** seconds. More than 100 collision vectors were required to span the entire damage width.* (Fatal)

EDSMAC divides the damage range into a number of discrete force vectors, RHO. The angular interval between RHO vectors is DELPSI, an input variable. This angular interval determines the number of RHO vectors required to span the damage range. Up to 100 RHO vectors are allowed. If the damage range is so wide that 100 RHO vectors are insufficient, this error occurs. To solve the problem, the value for DELPSI must be increased, thus allowing a wider damage range to be handled.

Message: *An error occurred during the collision phase at t = *.** seconds. After 200 bisections, the difference in intervehicle contact pressures was greater than the acceptable error, ALAMB.* (Fatal)

During the collision, force equilibrium is sought between each of the vehicle crush vectors, RHOBI and RHOBJ, using an iterative adjustment, DELRHO, of the vector lengths. Up to 200 adjustments of each vector are allowed. If, after 200 adjustments, force equilibrium is not reached (i.e., the difference in forces associated with the length of the vectors is still greater than ALAMB), this error is encountered. To solve the problem, the input value for ALAMB must be increased, or the input value for DELRHO must be reduced. Reducing the value of the integration time step, DTCOLL, may also help.

This error illustrates the problem when attempting to use EDSMAC for barrier collisions. By definition, the crush stiffness of a barrier is infinite. The value of AKV normally applied to a barrier would be 1×10^6. Each adjustment of the barrier's RHO would result in an enormous change in PRESJ (assuming the barrier is the *other* vehicle). Thus, equilibrium between PRESI and PRESJ will never be reached. After 200 attempts, this error message will be issued and further calculations will cease. One way of solving this problem is to use a lower value for the barrier stiffness, AKV, a smaller interval, DELPSI, between RHO vectors, a smaller adjustment, DELRHO, a larger maximum pressure error, ALAMB, and a smaller integration time step, DTCOLL. This method, however, has not been validated.

Message: *An error occurred while saving the acceleration or damage array maximum at t = *.** seconds.* (Fatal)

This message occurrs during SAVEMAX when the acceleration array has not been set for any damage range. This would indicate a logic error in the routine.

Message: *An error occurred while calculating a CDC for Vehicle *. The error code is **.* (Informative)

EDSMAC divides the damaged vehicle exterior into specific regions. Each region is associated with a certain entry of the vehicle's 7-character CDC. Specifically, columns 3-6 are determined by the location of damage.

This message indicates a logical error has occurred while attempting to assign the CDC. Virtually all these errors have been eliminated by verifying that valid vehicle dimensions were entered during the input session.

Message: *During the examination of individual damage ranges, no match was found for damage range No. *.* (Informative)

During the collision phase, the principal direction of force, PDOF, is computed from the resulting acceleration vector for each damage range. The angle of the middle of the damage range, PSIM, is also computed (simply the point between the beginning and ending angle for the damage range). If PSIM and PDOF are more than 60 degrees apart, this message is issued. The offending damage range can be identified by comparing the CDC clock direction with PSIM in the EDSMAC Damage Profiles (see **Graphics**).

This error message commonly occurs for sideswipe collisions, where the angle of the acceleration is nearly parallel to the damage surface. This error message used to be displayed inadvertently for some collision configurations. However, the problem in the original SMAC code has now been corrected in EDSMAC (see **Modifications**).

Message: *PRESI (or PRESJ) tends negative at t = *.** seconds.* (Informative)

During the collision phase, the lengths of each RHO vector, RHOBI for the base vehicle and RHOBJ for the other vehicle, are adjusted until equilibrium is reached between both vehicles. If the adjustment results in a RHOBI or RHOBJ vector which extends beyond the vehicle exterior, the associated contact pressure, PRESI or PRESJ, would be negative. Technically, this would suggest tension between the vehicles. Such a tension is not allowed to exist, and the contact pressure is set to zero.

Message: *The collision resulted in more than 10 individual regions of damage. EDSMAC limits the analysis to 10 regions.* (Fatal)

During the collision phase, the acceleration vs. time history is computed. The resulting curve may be smooth or it may have several peaks, depending on such events as secondary impact (side slapping) or major collisions involving significant dynamics. EDSMAC saves each acceleration peak for potential calculation of additional damage regions and associated PDOFs in the SAVEMAX routine. Up to 10 individual acceleration peaks may be saved. If the collision results in more than 10 peaks, the above message is issued.

DISCUSSION

The analytical basis for EDSMAC allows it to be used as an important and useful tool for illustrating how an accident may (or may not) have occurred. For example, suppose the occupants in a stationary vehicle believe their car was struck from the rear by a vehicle travelling 30 mph, yet the rear of their car is only slightly damaged. EDSMAC could be used to show what the damage to their car *should* look like if it were struck at 30 mph (see figures 16 & 17).

A characteristic all simulation programs, including EDSMAC, is the fact that a single simulation cannot be used to describe the *only* way an accident could have occurred. Experience has shown that it is easy to create different scenarios (sometimes significantly different!) by making small changes in the input data.

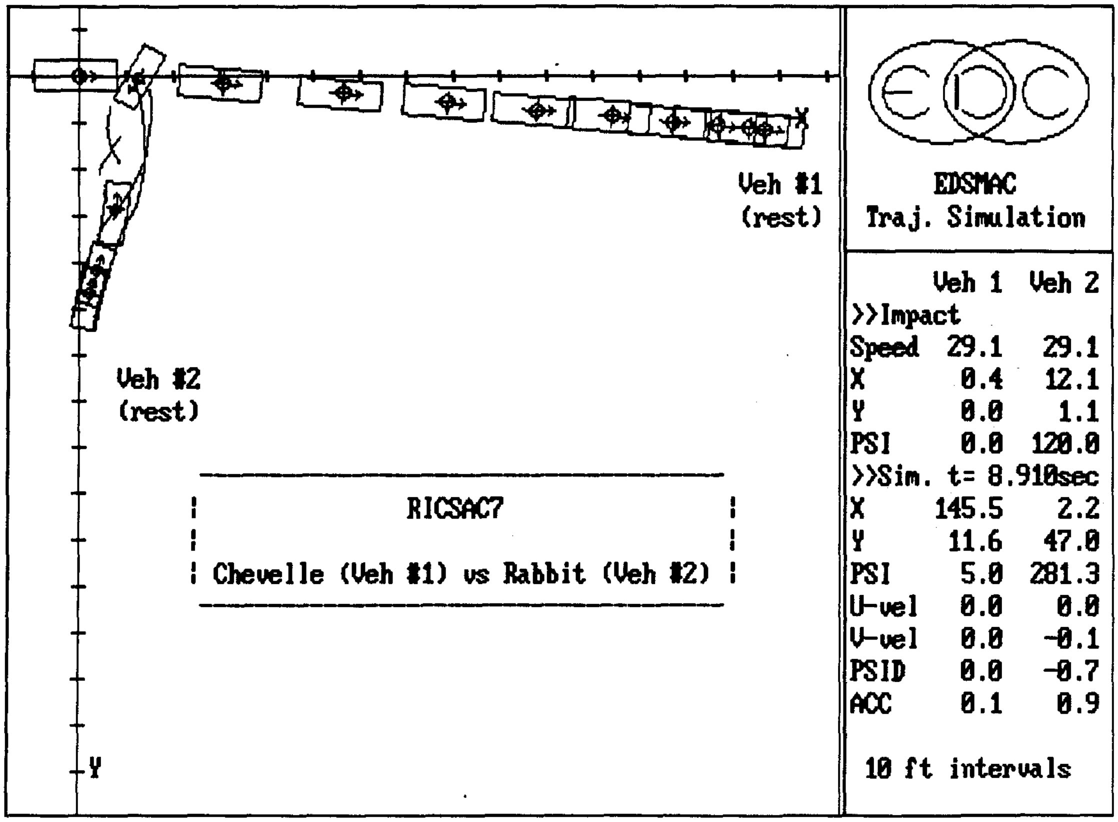


Figure 19 - EDSMAC results for RICSAC7 standard data set

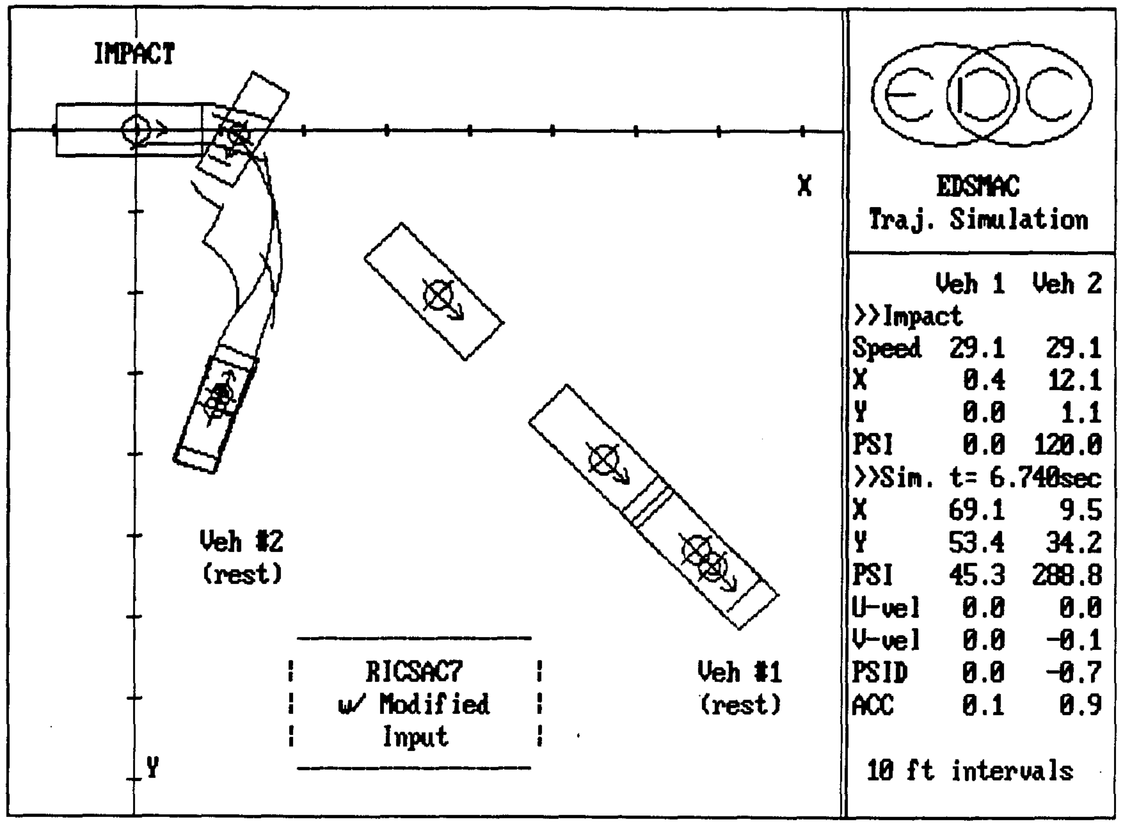


Figure 20 - EDSMAC results for RICSAC7 with a minor modification.

To illustrate this point, a typical EDSMAC input data set, RICSAC7 [11], was used to produce the trajectories shown in figure 19. Next, the inter-vehicle coefficient (AMU) was increased from 0.3 to 0.7 (well within a reasonable range for that variable). The result of this seemingly innocuous change is shown in figure 20.

The above observation must not be construed as a criticism of simulations. It should, however, create an awareness that reconstructions and simulations serve different purposes: Reconstructions attempt to find the only way an accident could have occurred, while simulations can identify many possible ways an accident could have occurred. This is true because an accident reconstruction based on good vehicle and accident site inspections has two additional laws of physics available: the laws of conservation of energy and momentum. Although a simulation satisfies these laws, it does not have the benefit of that information as input *before* the simulation begins.

CONCLUSIONS

1. EDSMAC uses initial conditions, vehicle properties, physics and numerical integration to simulate a collision and compute the delta-V.

2. EDSMAC was developed from the SMAC mainframe program, but utilizes input, output and graphics programs to take advantage of the personal computer environment.

3. Five modifications to the calculation procedures were introduced. Two of these affected the way the user interacted with the program: the braking/steering tables and the terrain boundary. The other three eliminated minor errors which affected some collision configurations and error messages.

4. Simulation and reconstruction methods analyze collisions using opposite, but complementary, approaches.

5. The calculation procedures for simulating a collision are simple in concept, but extremely complex in practice.

6. The program output included numerical results, graphics, and warning messages.

7. The EDSMAC graphics are an effective way to present the results in the form of vehicle trajectories and damage profiles.

8. The major benefit of EDSMAC is its ability to provide an analytical method for illustrating how an accident may, or may not, have occurred.

REFERENCES

1. Mackay, G.M., "Injury and Collision Severity", SAE Paper No. 680779, October, 1968.

2. Advances in Belt Restraint Systems: Design, Performance and Usage, Society of Automotive Engineers, Publication P-141, February, 1984.

3. Day, T.D., Nagle, D.G., Beals, R.K., "Case Studies Involving the Use and Non-use of Seatbelts", SAE Paper No. 861091, November, 1986.

4. Vehicle Analysis Package - EDSMAC Program Manual, Version 2, Engineering Dynamics Corporation, Lake Oswego, OR, 1986.

5. Solomon, P.L., "The Simulation Model of Automobile Collisions (SMAC) Operator's Manual", U.S. Department of Transportation, NHTSA, Accident Investigation Division, Washington, D.C., 1974.

6. Vehicle Analysis Package - EDCRASH Program Manual, Version 4, Engineering Dynamics Corporation, Lake Oswego, OR, 1986.

7. Vehicle Dynamics Terminology, Society of Automotive Engineers Technical Report SAE-J670e, July, 1976.

8. 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.

9. Fiala, E., "Seitenkrafte am Rollendum Luftreifen" (Lateral Forces on Rolling Pneumatic Tires), Zeitschrift V.D.I., 96, No. 29, 1954.

10. Collision Deformation Classification, Society of Automotive Engineers Technical Report SAE-J224 MAR80, March, 1980.

11. Jones, I., Baum, A., "Research Input for Computer Simulation of Automobile Collisions - Volume IV: Staged Collision Reconstructions", DOT HS-805 040, December, 1978.

12. McHenry, R., Jones, I., Lynch, J., "Mathematical Reconstruction of Highway Accidents - Scene Measurement and Data Processing System", DOT HS-053-3-658, December, 1974.

Positions and opinions advanced in this paper are those of the author(s) and not necessarily those of SAE. The author is solely responsible for the content of the paper. A process is available by which discussions will be printed with the paper if it is published in SAE Transactions. For permission to publish this paper in full or in part, contact the SAE Publications Division.

Persons wishing to submit papers to be considered for presentation or publication through SAE should send the manuscript or a 300 word abstract of a proposed manuscript to: Secretary, Engineering Activity Board, SAE.

Printed in U.S.A.