SAE TECHNICAL PAPER SERIES

1999-01-0102

An Overview of the EDSMAC4 Collision Simulation Model

Terry D. Day
Engineering Dynamics Corporation

Reprinted From: Accident Reconstruction: Technology and Animation IX (SP-1407)

SAE The Engineering Society For Advancing Mobility Land Sea Air and Space®
INTERNATIONAL

International Congress and Exposition
Detroit, Michigan
March 1-4, 1999

400 Commonwealth Drive, Warrendale, PA 15096-0001 U.S.A. Tel: (724) 776-4841 Fax: (724) 776-5760

ISSN 0148-7191

Printed in USA 90-1203B/PG

1999-01-0102

An Overview of the EDSMAC4 Collision Simulation Model

Terry D. Day
Engineering Dynamics Corporation

ABSTRACT

The EDSMAC simulation model has been in widespread use by vehicle safety researchers since its introduction in 1985. Several papers have been published that describe the model and provide validations of its use. In 1997, the collision and vehicle dynamics models were extended significantly. The main control logic was also extended and generalized. The resulting model was named EDSMAC4. This paper describes the EDSMAC4 model with particular attention to the extensions to the original algorithms. The paper also provides a validation of the new model by direct comparison to staged collision experiments and the results from the previous EDSMAC model. Extensions found in the new algorithm include the ability to simulate articulated vehicle collisions, simultaneous collisions between any number of vehicles (the original was limited to two vehicles), the adoption of a more realistic force deflection model using A and B stiffnesses (as opposed to a single K_V stiffness that assumes the force-deflection curve goes through the origin), the ability to simulate vehicles with tandem axles and dual tires, and vehicle longitudinal and lateral load transfers. The validation study found differences in the results when compared to the previous EDSMAC program. New EDSMAC4 data sets were produced that reflect the changes to the model.

THE EDSMAC COLLISION simulation model was introduced in May of 1985. Its calculation procedures and validations have been published in the literature [1,2,3]*. Other researchers have used EDSMAC's results as a standard basis for comparison between collision algorithms [4,5]. Since its introduction, more than 1000 safety researchers have used EDSMAC to study motor vehicle collisions.

The EDSMAC model was developed from the public domain SMAC program, produced by Calspan [6,7] for NHTSA [8]. The main difference between the EDSMAC and SMAC models was the addition of a user interface (SMAC was a batch-mode program developed for mainframe computers, while EDSMAC was an interactive, menu-driven program developed for the PC). Differences in the models were reported in references 1 and 2.

This paper reports on an extended version of the EDSMAC program, called EDSMAC4. These extensions resulted in significant changes to major portions of the original code. In particular, the *control routine logic* was revised to allow:

- simulation of any number of vehicles,
- simultaneous collisions with multiple vehicles, and
- improved collision detection.

The *collision algorithm* was extended with:

- an improved force-deflection model,
- different stiffnesses for front, back and sides,
- support for barrier crashes,
- damage profile simulation/visualization, and
- support for articulated vehicle crashes.

The *vehicle dynamics model* was extended with:

- support for tandem axles and dual tires,
- calculation of load transfers,
- support for wheel displacement during impact,
- support for the tire blow-out model, and
- support for articulated vehicles.

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

This paper describes in detail these additions to the mathematical/physical models as well as extensions to the general algorithms that embody these models. The paper also includes a validation study comparing EDSMAC and EDSMAC4 results with staged collision experiments.

PROGRAM DESCRIPTION

EDSMAC4 is an HVE-compatible [9,10,11,12] simulation model developed for use by motor vehicle safety researchers and crash investigators. The vehicles may be unit vehicles (e.g., passenger cars and pickups) or articulated vehicles (tractors towing up to three trailers). The crash sequence may be simulated on flat roads, irregular 3-D surfaces or anything in between.

Like all simulation programs, EDSMAC4 predicts the outcome of an event based on user-supplied initial conditions. These initial conditions are the initial vehicle positions and velocities. The user also supplies a set of time-based driver controls (steering, throttle and brakes) for each vehicle. The initial conditions may be at impact or before impact. If the simulation begins before impact, the user can include pre-impact driver avoidance maneuvers in the simulation.

Procedure For Using EDSMAC4

EDSMAC4 uses the HVE simulation environment (see Figure 1); thus, EDSMAC4 uses the same procedures as other HVE-compatible simulation models. These procedures are as follows:

- Use the HVE Vehicle Editor to select one or more vehicles from the vehicle database according to type, make, model, year and body style. If desired, edit the vehicle properties.
- Use the HVE Environment Editor to create the environment. This step includes assigning the location, time and date of the crash for positioning the sun, and building or importing a 3-D geometry file for the crash site environment.
- Use the HVE Event Editor to create (choose the vehicles and EDSMAC4 calculation model) and set up (assign the initial vehicle positions, velocities and driver controls) the simulation. If desired, assign target positions (instances of each vehicle placed along the desired trajectory to assist in determining if the vehicles are following the measured path; target positions are not used by the calculation), tire blowout and wheel displacement parameters, and accelerometer locations for each vehicle.

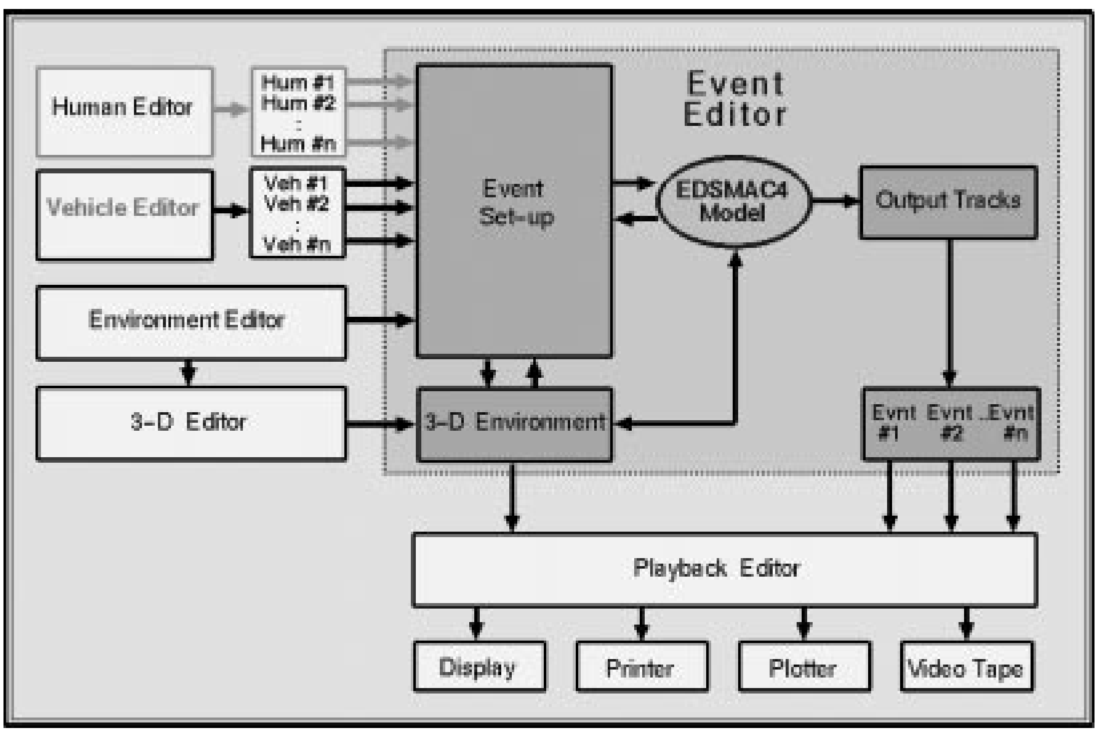


Figure 1 - HVE Simulation Environment, with EDSMAC4 as the current simulation model in the HVE Event Editor

- Use the HVE Event Editor to execute the event. After execution, compare the simulated path with the measured path (using target positions, if entered). If necessary, adjust the vehicle initial positions, velocities and driver controls as required to achieve a match between the simulated and measured paths and damage profiles.
- Use the HVE Playback Editor to produce the desired numeric, graphic and video reports. The Playback Editor is also used to combine the EDSMAC4 results with results from other simulations into a seamless sequence involving multiple events.

GENERAL CALCULATION PROCEDURES

EDSMAC4 (like EDSMAC) uses two major calculation procedures during the processing phase. These procedures are (1) the *Main Simulation Routine* and (2) the *Damage Summary*. These two procedures are described in the following sections.

Main Simulation Routine

Most of the EDSMAC4 program control is performed by the *main simulation routine*. The main simulation routine (see Figure 2) consists of the main simulation *control loop* and five major functions. (In the *C* programming language, a *function* is similar to a FORTRAN *subroutine*.) These functions are executed from within the main simulation control loop at each successive time increment until a termination condition occurs (e.g., both vehicles are stopped or the maximum simulation time is reached). The control loop and major functions and procedures are described below.

Control Loop – The main simulation control loop (see Figure 3) controls execution using five important functions. On termination, the control loop passes control back to the main simulation routine.

The first function used by the control loop is `RungeKutta()`, the numerical integration function used at

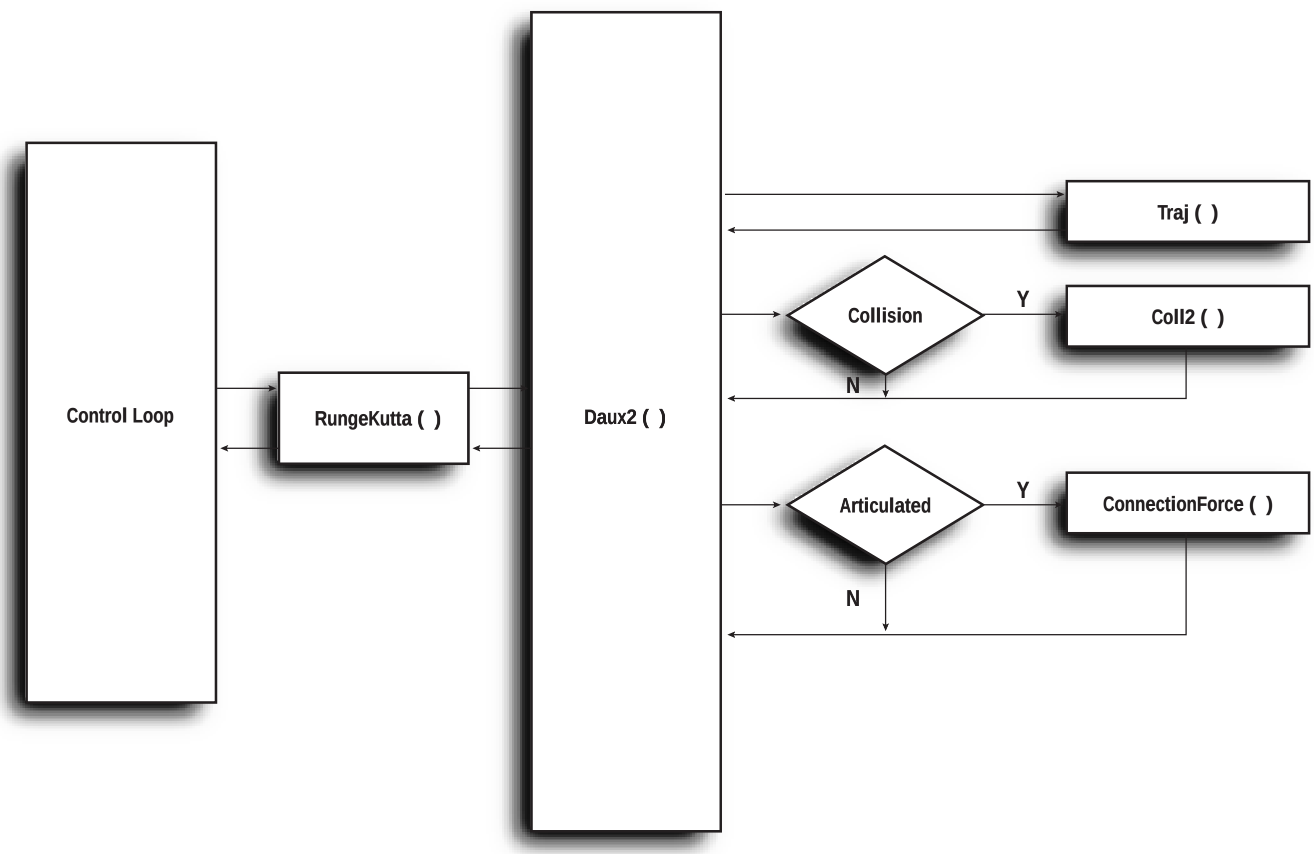


Figure 2 - EDSMAC4 Main Simulation Routine flow chart, illustrating the Control Loop and important functions. This flow chart describes the overall program execution procedure.

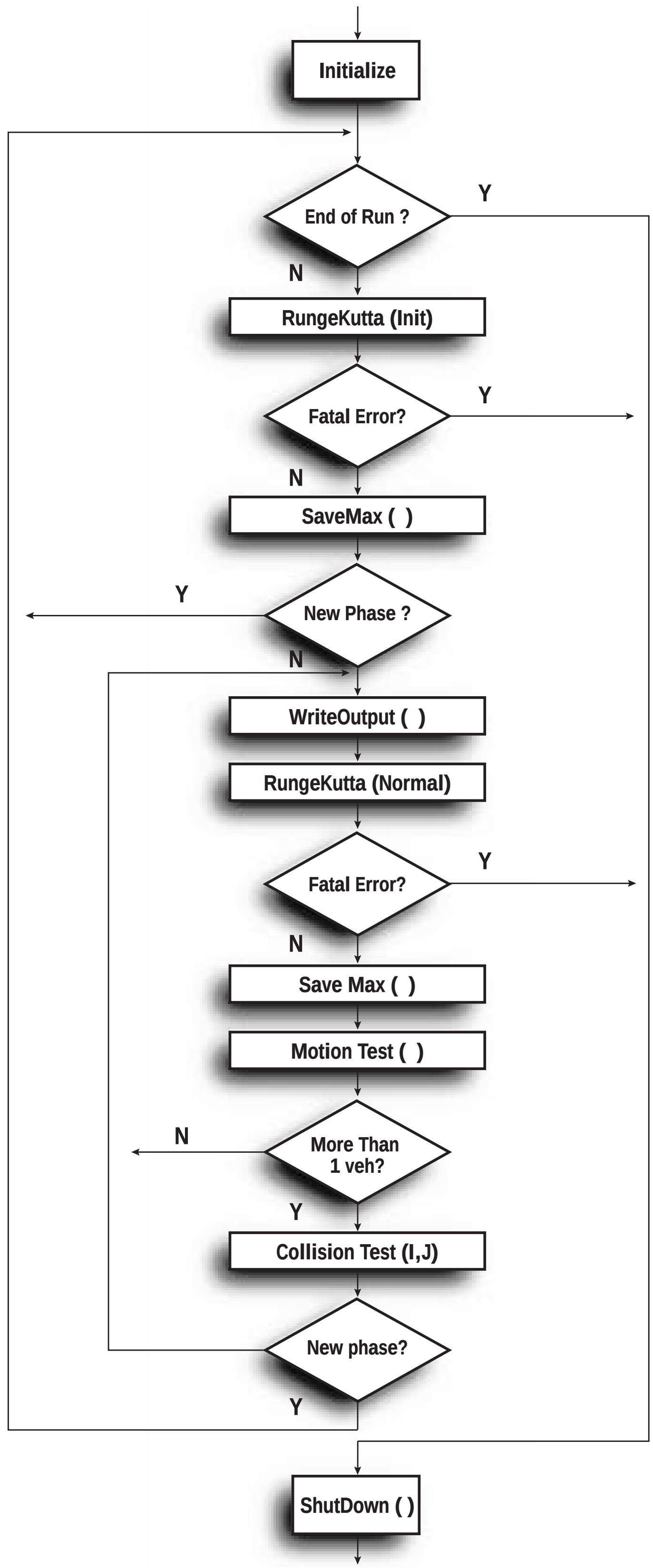


Figure 3 - EDSMAC4 Control Loop flow chart

each timestep. EDSMAC4, like EDSMAC, uses a fixed timestep, fourth-order, Runge-Kutta numerical integration method. The numerical integration function calls another function, named `Daux2()`, which, in turn, calls the routines that calculate all forces and moments acting on each vehicle. `Daux2()` then calculates the accelerations and returns control to `RungeKutta()`, where the vehicle positions and velocities are updated for the next time step.

A function called `SaveMax()` evaluates the motion that occurred during the current time step and compares the accelerations with those that occurred during the previous timestep. `SaveMax()` saves the magnitude and direction of the individual peak accelerations for each clock direction of acceleration for each vehicle. As will be shown later, the delta-V for each clock direction is calculated from these stored peak accelerations.

The next function in the control loop is called `WriteOutput()`. This function is executed whenever the current simulation time has reached the specified print time interval, and causes the vehicle kinematics (positions, velocities and accelerations), kinetics (summation of tire, inter-vehicle connection, and collision forces and moments acting on each sprung mass), accelerometer results, damage profile, tire forces, wheel positions, inter-vehicle connection forces and driver controls to be written to the HVE output tracks (see Figure 1; Event Editor).

A function called `CollisionTest()` determines if any two vehicles are in contact with each other. If true, the collision flag is set for this pair of vehicles, and the time of their collision is stored for display in the output reports.

If the simulation time has reached the maximum time specified by the user, or if *all* vehicles have stopped moving (as specified by the termination velocities), the main simulation loop calculations stop and control is passed to the damage summary routine. The main simulation routine is also terminated by a fatal error (e.g., *Too Many Damage Vectors*).

Tire Force Computation – The `Traj()` function computes the forces at each tire during the current time interval. The Fiala tire model [13] and 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 according to the current vertical tire load, friction, surface normal, tire slip angle and cornering stiffness.

Collision Force Computation – The `Coll2()` function computes the collision forces and moments at the current time interval. Details of the procedure are provided later in this report (see **Collision Force**).

Connection Force Computation - If the simulation involves one or more articulated vehicles, the `ConnectionForce()` function computes the forces and resulting moments from the inter-vehicle connections during the current timestep.

Acceleration Computation – The `Daux2()` function sums the forces and moments from the tire, collision and inter-vehicle connections at the current timestep. `Daux2()` then uses these total forces and moments to calculate the current level of linear and angular acceleration for each vehicle.

Main Control Logic Extensions

The main control logic was rewritten for EDSMAC4. The original FORTRAN code, which was rather sparsely commented and difficult to follow, was re-organized using a structured programming paradigm (the code had already been rewritten using the C language, as reported in reference 3). The result was a robust code that is easy to follow, easy to maintain and easy to extend.

After rewriting the main control routine, the following extensions were added:

Number of Vehicles - The code was generalized to allow the simulation of any number of vehicles, instead of just two. This change allows the direct study of chain reaction collisions. Simulation of events involving multiple non-contacting vehicles is also simplified. The change was implemented through the use of a *vehicles list*, and maintaining a history of contact and separation for each pair of vehicles.

Simultaneous Collision -The basic design of the new code inherently allows any vehicle to be in a collision with any number of other vehicles at the same time. Thus, a vehicle sandwiched between two other vehicles during a rear-end collision is properly handled (this capability also relies on the extensions to the collision algorithm, described below).

Collision Detection - The `CollisionTest()` function includes new code that decreases the collision integration timestep one timestep before collision occurs, instead of at the first sign of overlap. This change prevents excessive vehicle overlap at the start of the collision phase, and resulting fatal error (previously, EDSMAC users solved this problem by manually reducing the time steps used for the trajectory and separation phases).

Damage Summary

The *damage summary* routine is executed after the conclusion of the main simulation routine. The damage summary routine includes two procedures: the `DamageRange()` and the `CDC()` functions. These functions are executed 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, described next. The `DamageRange()` function analyzes and organizes the vectors to determine the number of individual damage locations, or *damage ranges*, on the vehicle exterior.

CDC – Each individual damage range is compared with the original vehicle dimensions to compute a CDC (the CDC is a 7-character code describing the vehicle damage [14]. The `CDC()` function performs this task through a series of logical comparisons between each damage range computed by the `DamageRange()`function and the damage limits corresponding to 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 is RICSAC Validation Test No. 9, found later in this report. During the intersection-type collision, 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.

COLLISION FORCE

The collision forces and moments are calculated by the `Coll2()` function. `Coll2()` computes the inter-vehicle force between vehicles *I* and *J* at discrete time intervals specified by the numerical integration process (*I* and *J* are the indices of any two vehicles in a vehicle list of length *NumVehicles*). The flow chart in Figure 4 describes the process for calculating these forces and moments. The steps are described below.

Setting up the Relative Coordinates with Vehicle I as Base - Initial 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 *I* to define the reference frame for collision calculations (see figure 5). At this point in time, a set of temporary indices, *I'* and *J'*, are defined. The index, *I'*, identifies the *base* vehicle (i.e., the vehicle from which the reference frame is defined, and the index, *J'*, identifies the *other* vehicle.

As shown in Figure 5, vehicle *J* can be precisely located from vehicle *I*'s current earth-fixed X,Y coordinates and heading angle. Because vehicle *J*'s exterior dimensions are also known, two angles, *PSIBPB* and *PSIBPF* can be established. Logically, any damage to the surface of the base vehicle 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 angles *PSIBPB* and *PSIBPF*. Each *RHO* vector represents a potential interaction point on the vehicle. The angle between each vector on the base vehicle is established by the value of *delPSI*, an

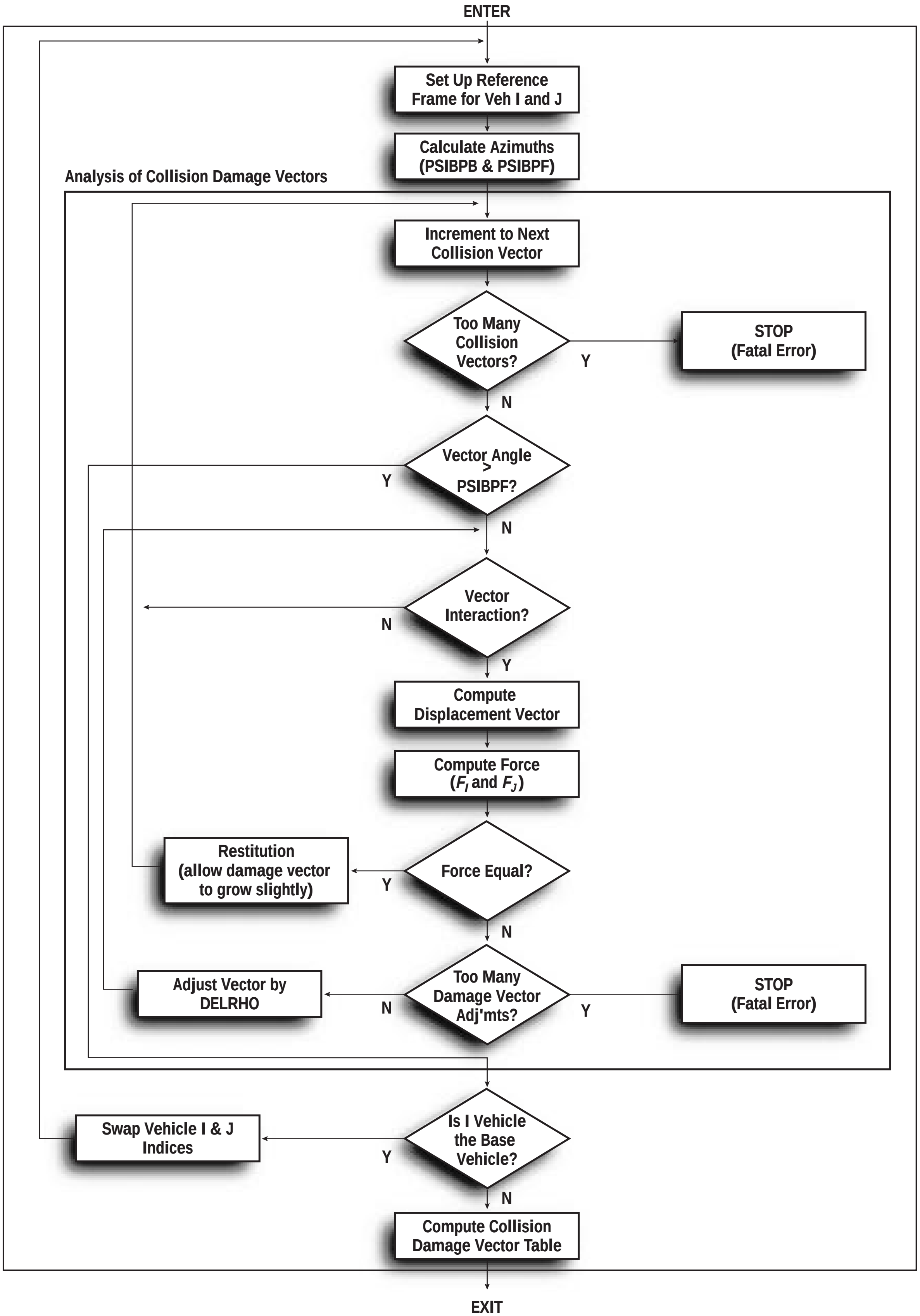


Figure 4 - COLL2() flow chart, used for calculating the collision forces and moments during each timestep.

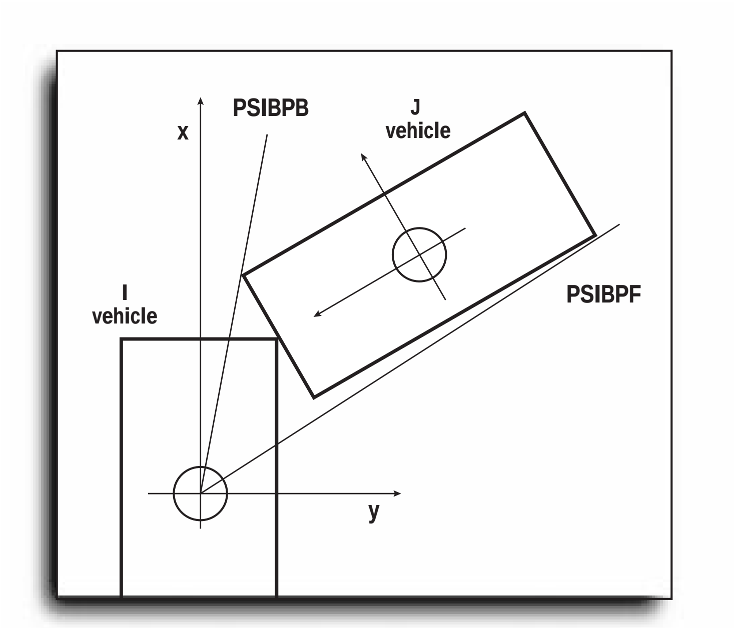


Figure 5 - Viewing vehicle J using vehicle I as a reference frame.

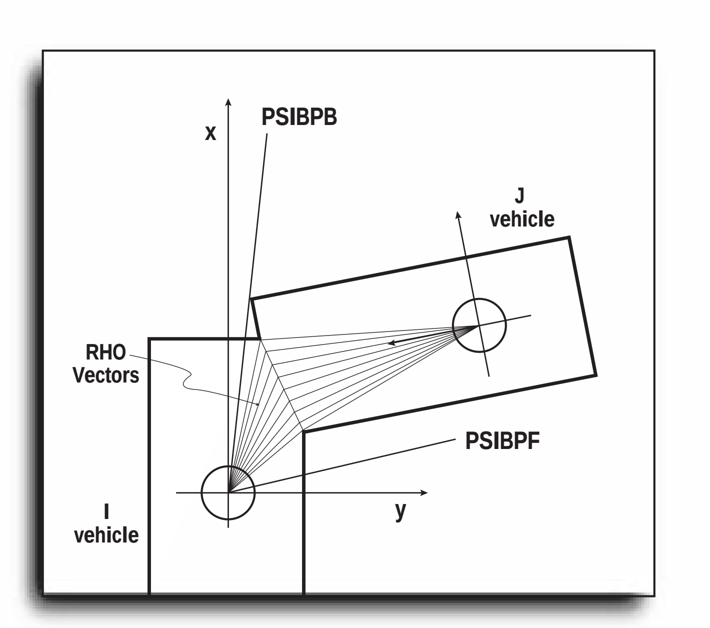


Figure 6 - Rho vectors

event input parameter (usually about 2 degrees). The angle on the other vehicle is almost always different from the base vehicle because, while that 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 force) between the vehicles is confirmed when the end of a *RHO* vector on 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. These points provide the basis for what will later become the damage profile.

Computing the Displacement Vector - Once interaction for a particular *RHO* vector is confirmed, two additional vectors are established (see Figure 8). RHO_I is a vector from the base vehicle CG to the interaction point and RHO_J 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.

The distance from the original vehicle perimeter to a new interaction point specifies a change in the length of the RHO_I and RHO_J vectors. This interaction point represents a location on the vehicle where the inter-vehicle force is computed.

Force-Deflection Model

The force vs deflection model in the original SMAC and EDSMAC models is

$$F = K_v \delta$$

where F is the force on a *RHO* vector, K_v is the spring constant (or stiffness) of the *RHO* vector and δ is the deformation of the *RHO* vector from its original length. As reported during the validation of EDSMAC [2], this model results in over-estimation of the simulated crush depth for collisions wherein a vehicle's delta-V was less than about 30 mph; the problem is greater for low-speed crashes. Note the authors also pointed out this problem had little, if any, effect on trajectories. However, it may be important if an EDSMAC collision pulse is used in an occupant simulation.

In EDSMAC4, the force vs deflection model has been changed to

$$F_n = A_n \xi_n + B_n \delta_n$$

where F_n is the force on RHO_n , A_n is the initial deflection constant, ξ_n is a null distance for reaching the full value of A_n, B_n is the deformation constant (i.e., spring stiffness) of RHO_n during deformation, and δ is the simulated deformation of RHO_n. These A and B constants are the same as those used by the EDCRASH damage algorithm. Note this force vs deflection relationship is essentially the same as the relationship defined

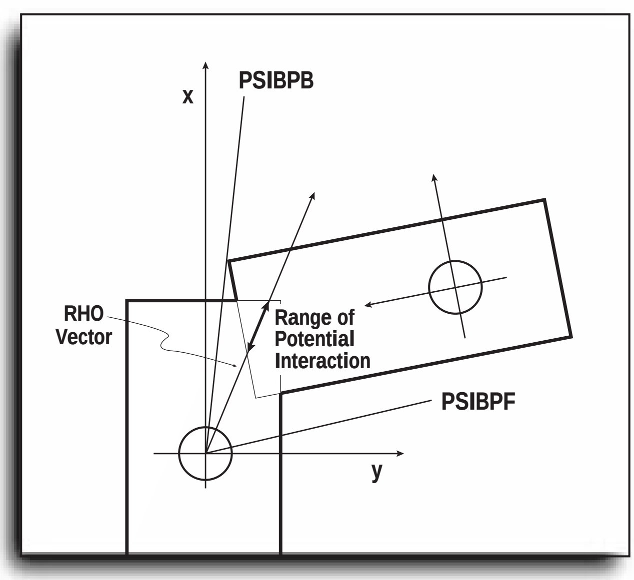


Figure 7 - Seeking interaction. A force on a Rho vector exists if its endpoint lies inside the other vehicle.

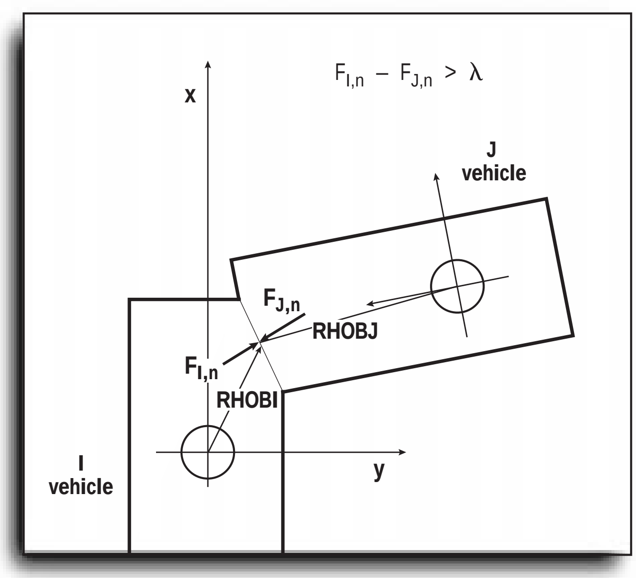


Figure 8 - Displacement vector

by Campbell [15]. The revised model (see Figure 9) is a more realistic model of a vehicle exterior, and should result in an improved estimation of crush depth, acceleration magnitude and collision time interval (see also reference 2).

During execution of the `Coll2()` function, F_n is calculated for each *RHO* vector, *n*, on each vehicle, *I* and *J*. Force equilibrium is maintained by confirming

$$F_{I,n} = F_{J,n} \pm \lambda$$

where λ is a user-assigned allowable force imbalance, normally about 15-20 lb/in. If the difference is less than λ, then $F_{I,n}$ is set equal to $F_{J,n}$. If the difference is greater than λ, the interaction point is moved along vector $RHO_{I,n}$ by an increment, *delRHO*, a user-adjustable parameter, usually about 0.2 inches. From the new interaction point, $\delta_{I,n}$ and $\delta_{J,n}$ are recomputed. The new values of $F_{I,n}$ and $F_{J,n}$ are compared with λ for equality. If the forces are still unequal, the interaction point on $RHO_{I,n}$ vector is moved again. This process continues until $F_{I,n}$ and $F_{J,n}$ are within the allowable difference, λ. The interaction point can be moved up to 3000 times in an effort to equalize $F_{I,n}$ and $F_{J,n}$. If more than 3000 adjustments occur, a fatal error is issued and calculations stop.

After $F_{I,n}$ and $F_{J,n}$ are equalized, the next *RHO* vector, RHO_{n+1}, is analyzed using the above process. Up to 360 *RHO* vectors are allowed for each vehicle (i.e., n_{max}=360), 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*, and the pressure, on each *RHO* vector. The endpoints for the table of Rho vectors for each vehicle represents its damage profile.

If RHO_I has been previously computed, but the end of RHO_J falls outside vehicle *I*'s perimeter, the point is called a *J* point. In this case, the point on the damage profile of vehicle *I*, is calculated from vehicle *J*.

Restitution - Elastic restitution is modeled by allowing the length of the *RHO* vector to increase slightly after each timestep. The amount of the increase is modeled by a 2nd order polynomial based on user-entered polynomial coefficients, C_0, C_1, C_2. The amount of increase is a function of each *RHO* vector's initial length and current deformation. The effect of increasing the length of the *RHO* vector is that an additional force is computed that tends to push the vehicles apart, resulting in restitution. The restitution algorithm is bypassed for J-points (see preceding paragraph) because any lengthening of the *RHO* vector might result in a damage point outside the vehicle perimeter. The restitution algorithm is also bypassed if C_0=C_1=C_2=0.

Swap and Repeat - The above calculations use vehicle *I* as the base vehicle. The next step is to swap reference frames and repeat the calculations using vehicle *J* as the base vehicle, as shown in figure 10.

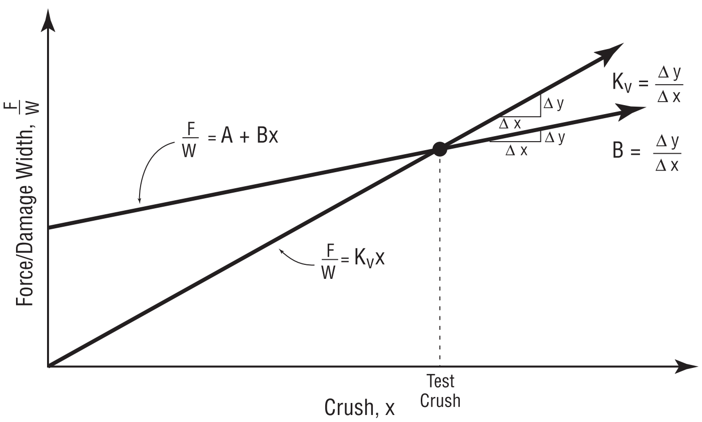


Figure 9 - EDSMAC4 force-displacement model

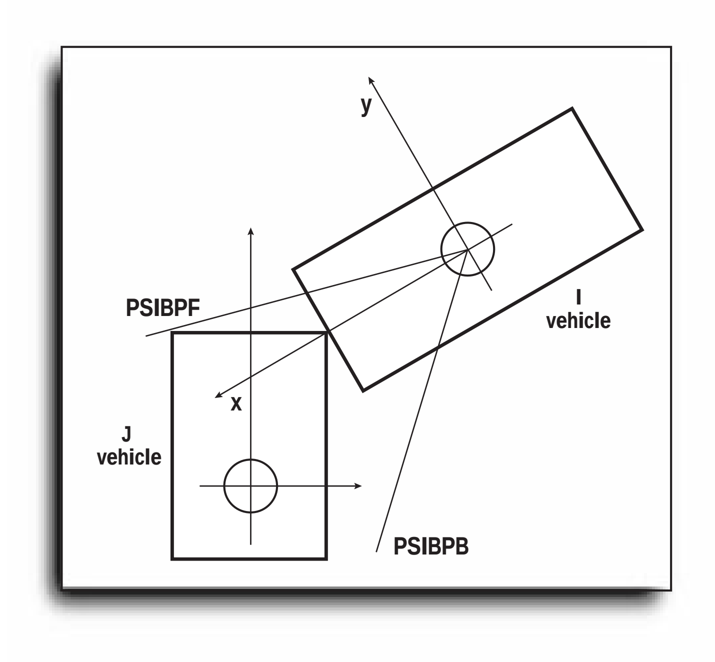


Figure 10 - Swap and repeat

The damaged perimeter of each vehicle changes as it is damaged. The current 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 Collision Force

The damage table provides up to 360 individual force vectors applied to each vehicle mass at a known location on the damage profile during the current time step. The final procedure it 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 11) during a given integration time step. These vehicle-fixed forces and moments are:

$$F_x = \sum_{n=1}^{n_{max}} F_{x,n_{coll}}$$

$$F_y = \sum_{n=1}^{n_{max}} F_{y,n_{coll}}$$

$$M_z = \sum_{n=1}^{n_{max}} F_{x,n_{coll}} r_{x,n} + F_{y,n_{coll}} r_{y,n}$$

where n_{max} is the number of *RHO* vectors and r is the vehicle-fixed location where F_x,F_y is applied.

Delta-V

Velocity is the time integral of acceleration, $dV = \int a dt$. Graphically, this is simply the area under the acceleration vs. time curve (figure 12). During the collision phase, the incremental areas for each time step are summed by using the average acceleration for the current time increment (figure 12). These accelerations are stored by `SaveMax()`. The delta-V is equal to the total area under the acceleration vs time curve at the conclusion of the collision phase.

The vehicle-fixed 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 clock direction, a unique delta-V is determined for each CDC having a different clock direction.

Extensions to the Collision Algorithm

The collision algorithm was completely rewritten for EDSMAC4 (the code had already been rewritten using the C language, as reported in reference 3). The result was a robust code that is easy to follow, easy to maintain and easy to extend.

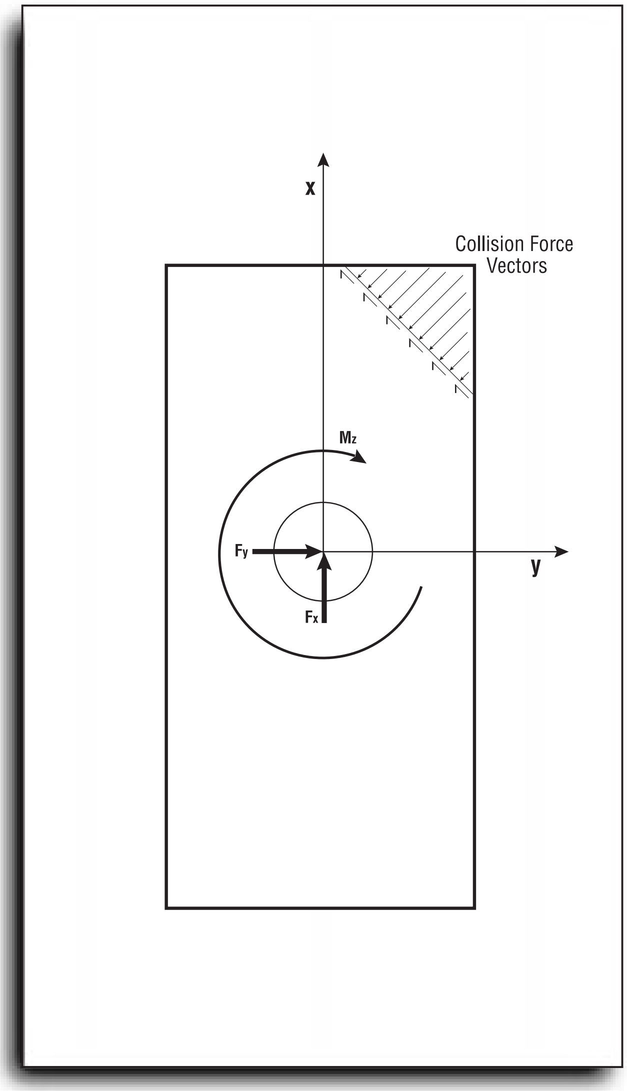


Figure 11 - Summing the collision forces and moments

After rewriting the collision algorithm, the following extensions were added:

Barrier Collision Simulation - The basic nature of the SMAC collision algorithm depends on the presence of vehicle deformation. Since barriers by definition do not deform, the algorithm is not directly amenable to barrier collisions, as stated in reference 16. The algorithm often terminates with a collision phase error (unable to balance forces or too many collision vectors). However, EDSMAC has frequently been used with great success, providing the collision is set up properly. In general, this requires ensuring the vehicle and barrier do not overlap by a large amount during the first collision timestep.

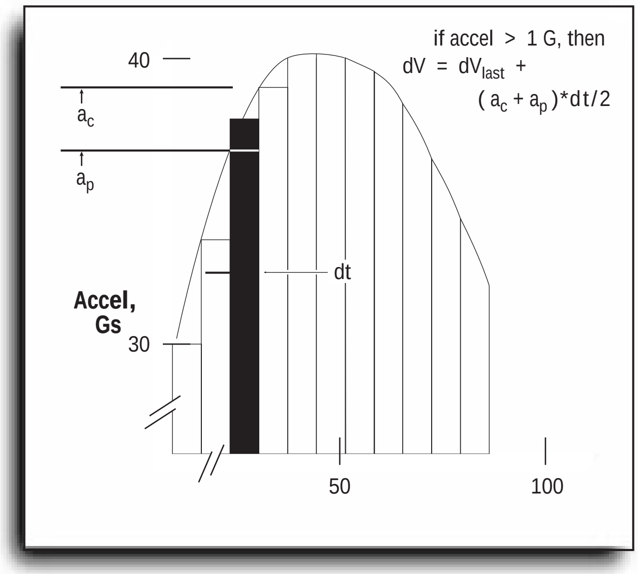


Figure 12 - Acceleration vs time history. Delta-V is the area under the curve.

Two extensions to EDSMAC4 greatly improve its ability to handle barrier collisions. First, the new proximity detection algorithm prevents excessive overlap during the first collision timestep. In addition, collision vector adjustment increment, *delRHO*, is reduced by a factor of 10 and the number of allowed interactions is increased by a factor of 10. Tests show these changes have been effective.

A third extension was added to help in the simulation of pole (narrow object) impact. The number of allowable collision damaged vectors was increased to 360 from 100. This allows the entire vehicle perimeter to be damaged; the algorithm does not allow penetration beyond the CG.

Articulated Vehicle Simulation - For articulated vehicles, an additional task is required of the collision algorithm, that is, to allow the trailer and vehicle to overlap without calculating a force. Further, it is required that the collision algorithm begin calculating a force between the tow vehicle and trailer when the yaw articulation angle exceeds that required for pinching between the body of the tow vehicle and body of the trailer. Given a user-entered maximum articulation angle, the connection location and the vehicle exterior dimensions, the *RHO* vectors that are in the overlapped region can be determined and ignored in the calculation of inter-vehicle forces. This procedure is illustrated in Figure 13.

Damage Profile Simulation - The HVE simulation environment provides the vehicle's 3-D mesh as input to a simulation model [9,11]. The mesh has both visual and

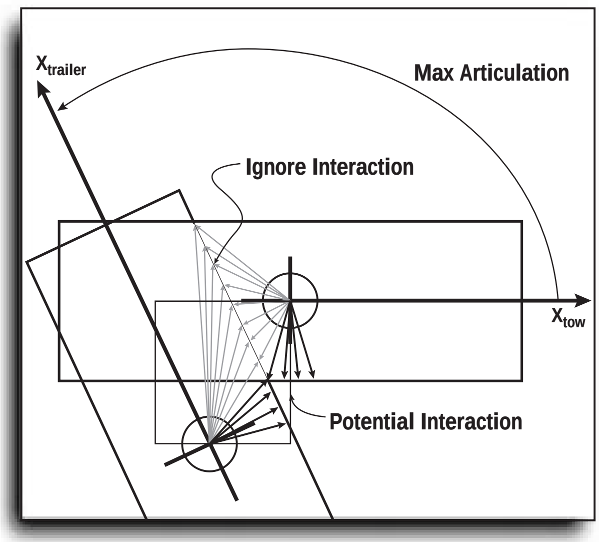


Figure 13 - Modeling collision force for articulated vehicles. Rho vectors inside the pinch point are ignored.

mechanical attributes. The visual attributes (e.g., 3-D surface geometry, color, transparency) are used to visualize the vehicle. The mechanical attributes (e.g., force-deflection relationship and friction for each surface vertex) provide the ability for collision simulations to calculate forces acting between vehicle meshes for two or more vehicles. Because EDSMAC4's collision algorithm is 2-dimensional, it cannot use the mesh mechanical attributes as input, however, it can use the mesh 3-D geometry for output. That is, the mesh can be used to visualize the damage predicted by the EDSMAC4 collision algorithm.

A procedure, called `MakeProfile()`, has been developed that uses the current *RHO* vectors produced by the `Coll2()` function to calculate the vertex displacements on the vehicle mesh. The resulting damage profile is automatically displayed in HVE's 3-D viewers.

The basic approach used by `MakeProfile()` is to find all vehicle mesh vertices in the vicinity of a given *RHO* vector, and move the x,y coordinates of those vertices to towards the x,y coordinates of the *RHO* vector. Linear interpolation is used for vertices between *RHO* vectors. Because the `Coll2()` procedure is 2-dimensional, the z-coordinate of the vertex is not changed. A flow chart for `MakeProfile()` is shown in Figure 14, and an example of a vehicle with a side damage profile is shown in Figure 15. See also Figures 18, 19, 21 and 22 for vehicle damage profiles in the validation study. Although this procedure is rather simple, it appears to do a very good job of allowing the user to visualize the EDSMAC4-simulated damage profile.

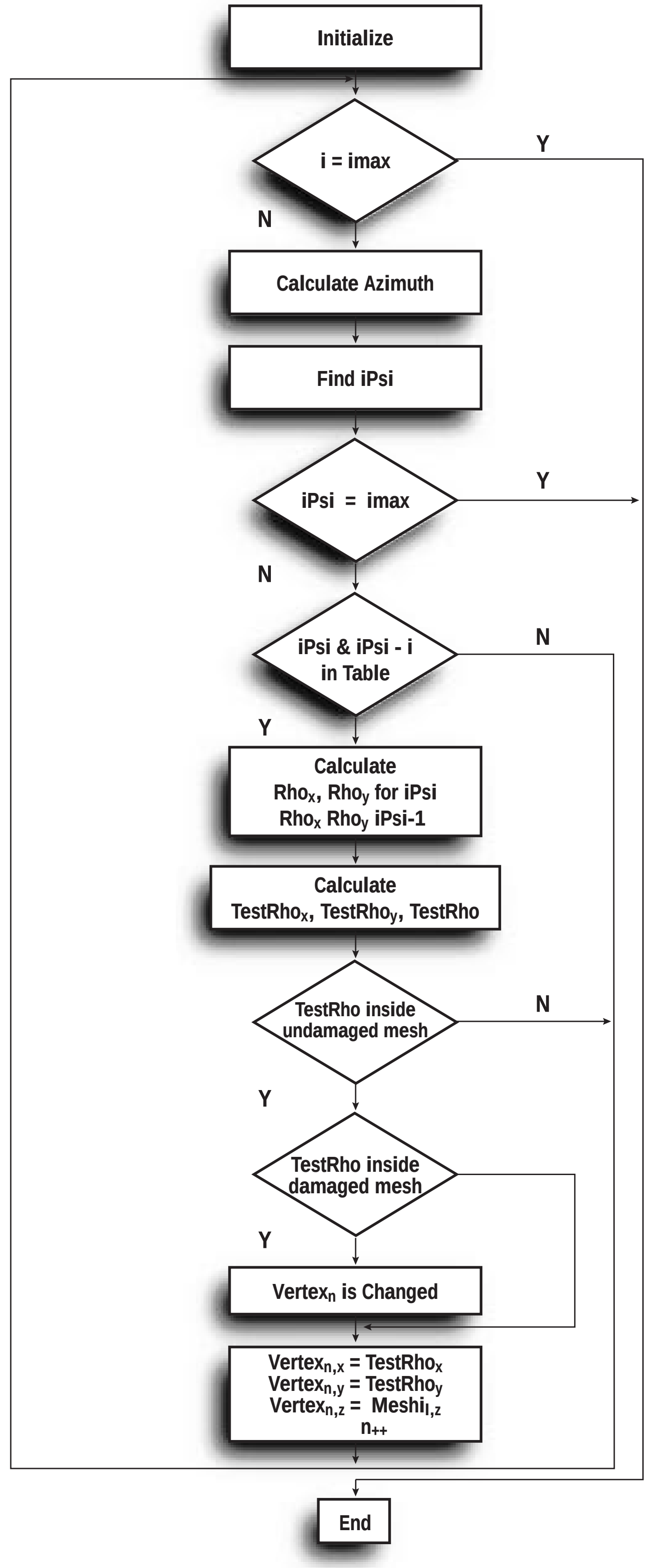


Figure 14 - Flow chart for damage profile visualization

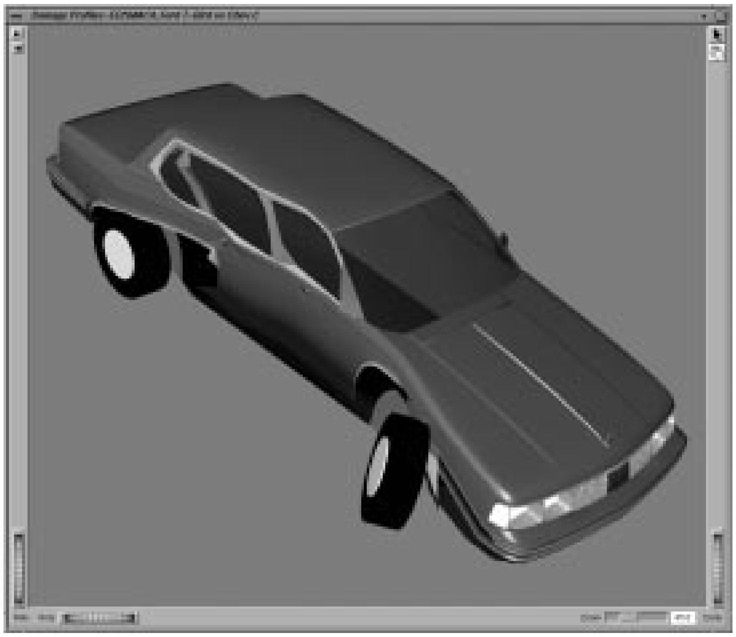


Figure 15 - Example of damage profile visualization using the MakeProfile algorithm (see also Figure 14).

VEHICLE DYNAMICS MODEL

The EDSMAC4 vehicle dynamics model is a 3-degree-of-freedom (per vehicle), simulating X-Y translation and yaw rotation. Motion in the Z direction, and roll and pitch orientations, are calculated on a quasi-static basis.

The vehicle dynamics model performs the following procedures:

- Calculate Tire Slip Angles
- Calculate Vertical Tire Load
- Calculate Tire Force, $F_{x'}$, $F_{y'}$
- Summation of Tire Forces and Moments at CG

These procedures are described below.

Slip Angles - Slip angles are calculated from the wheel's current forward and lateral velocities:

$$\alpha = \text{atan2}(v_{wheel}, u_{wheel})$$

where α is the tire slip angle and u_{wheel} and v_{wheel} are the forward and lateral wheel velocities.

Vertical Tire Load - In EDSMAC4, the total vertical tire load is the sum of the static vertical load, the load from longitudinal load transfer and the load from lateral load transfer. The total vertical tire load is calculated from the vehicle weight, wheel positions, CG height and current longitudinal and lateral accelerations. The EDSMAC4 vehicle model does not include suspension characteristics. Therefore, the vehicle's roll couple distribution is used to distribute lateral forces between the front and rear axles.

Tire Forces - EDSMAC4, like EDSMAC, uses the friction circle and Fiala tire model to calculate tire longitudinal and lateral forces. The tire force calculations use as input the tire *cornering stiffness, slip angle, vertical load, slide friction* and *attempted longitudinal force* returned by linear interpolation of the driver brake and throttle tables. Output from the tire model is $F_{x'}$,$F_{y'}$ in the wheel-fixed coordinate system.

Forces and Moments at CG - After performing the tire force calculations for each tire, the tire forces are summed in the vehicle-fixed x and y directions, and moments are summed about the vertical z axis

$$F_{x,tires} = \sum_{i=1}^{MaxWheels} F_{x',i}\cos(\delta_i) - F_{y',i}\sin(\delta_i)$$

$$F_{y,tires} = \sum_{i=1}^{MaxWheels} F_{x',i}\sin(\delta_i) + F_{y',i}\cos(\delta_i)$$

$$M_{z,tires} = \sum_{i=1}^{MaxWheels} F_{y,i}x_{wheel,i} - F_{x,i}y_{wheel,i}$$

where $F_{x,i}$ and $F_{y,i}$ are returned from the tire model, x_{wheel} and y_{wheel} are the vehicle-fixed wheel coordinates, and δ_i is the vehicle-fixed steer angle. These values are used by `Daux2()` function, along with the collision forces and moments, to determine the total forces and moments acting on the vehicle at the current timestep.

Extensions to Vehicle Dynamics Model

When EDSMAC was originally ported to HVE, its tire model was extended to use HVE's `GetSurfaceInfo()` function. This function allows EDSMAC's tire model to consider the 3-D environment data. At each simulation timestep, the EDSMAC tire model queries the environment to learn the surface elevation, surface normal and friction beneath each tire. Thus, the tire forces are appropriately assigned according to the surface elevation, friction and slope.

As part of the extensions to EDSMAC4, the following extensions were added to the vehicle dynamics model:

Tandem Axles, Dual Tires - The original SMAC and EDSMAC algorithms assume the vehicle has 2 axles. 3-axled vehicles (i.e., trucks with tandem axles) and dual tires were not supported. Users normally address this issue by multiplying the cornering stiffness according to the number of tires and using an average wheelbase that accounted for the additional axle. EDSMAC4 was extended to directly support vehicles with tandem axles and dual tires.

Load Transfers - SMAC and EDSMAC ignore longitudinal and lateral load transfers. During normal braking, accelerating and steering, these load transfers are rather small. However, during and just following a collision, load transfers can be significant. EDSMAC4 has been extended to consider these load transfers.

Calculation of the current vertical tire load, F_Z, for each wheel location is

$$F_z = F_{z,static} + \Delta F_{z,long} + \Delta F_{z,lat}$$

where

$F_{z,static}$ = static vertical load (calculated once, during initialization)

$\Delta F_{z,long}$ = load transfer due to longitudinal acceleration (calculated at each timestep)

$\Delta F_{z,lat}$ = load transfer due to lateral acceleration (calculated at each timestep)

Articulated Vehicle Simulation - Simulation of articulated vehicle crashes required the additional capability of modeling of inter-vehicle constraint forces and moments. Details of the modeling procedure are provided in reference 17. In general, the connections are modeled as a spring/dash pot force produced at each vehicle's connection point(s). EDSMAC4 does not simulate the roll degree-of-freedom, thus, roll moments are not included in the simulated connection. The articulation model also incorporates inter-vehicle pinch forces and moments, described earlier.

Wheel Displacement - Wheels are often displaced during a crash. Because wheel locations determine the moments from tire forces in the yaw plane, wheel displacement can affect the amount of simulated vehicle yaw rotation. EDSMAC4 uses HVE's *Wheel Displacement* option to allow the user to dynamically reposition one or more wheels during a crash. Vehicle-fixed x,y wheel location and wheel camber are all updated as a function of time.

Tire Blow-out -Tires may also become deflated during a crash, normally as a result of rim damage. EDSMAC4 incorporates a limited version of the HVE Tire Blow-out Model. To incorporate the full version requires a fully 3-dimensional vehicle and tire model, as described in reference 18; EDSMAC4's blow-out model includes only a time-dependent change to the tire cornering stiffness at one or more wheel locations.

VALIDATION

The EDSMAC4 model has been validated using the RICSAC staged collisions [19], a set of twelve well-instrumented 2-car collisions conducted by Calspan. Differences between the EDSMAC and EDSMAC4 collision models result in differences in the simulation output. The differences are not great, however, for completeness a set of RICSAC input data sets for EDSMAC4 have been produced [20].

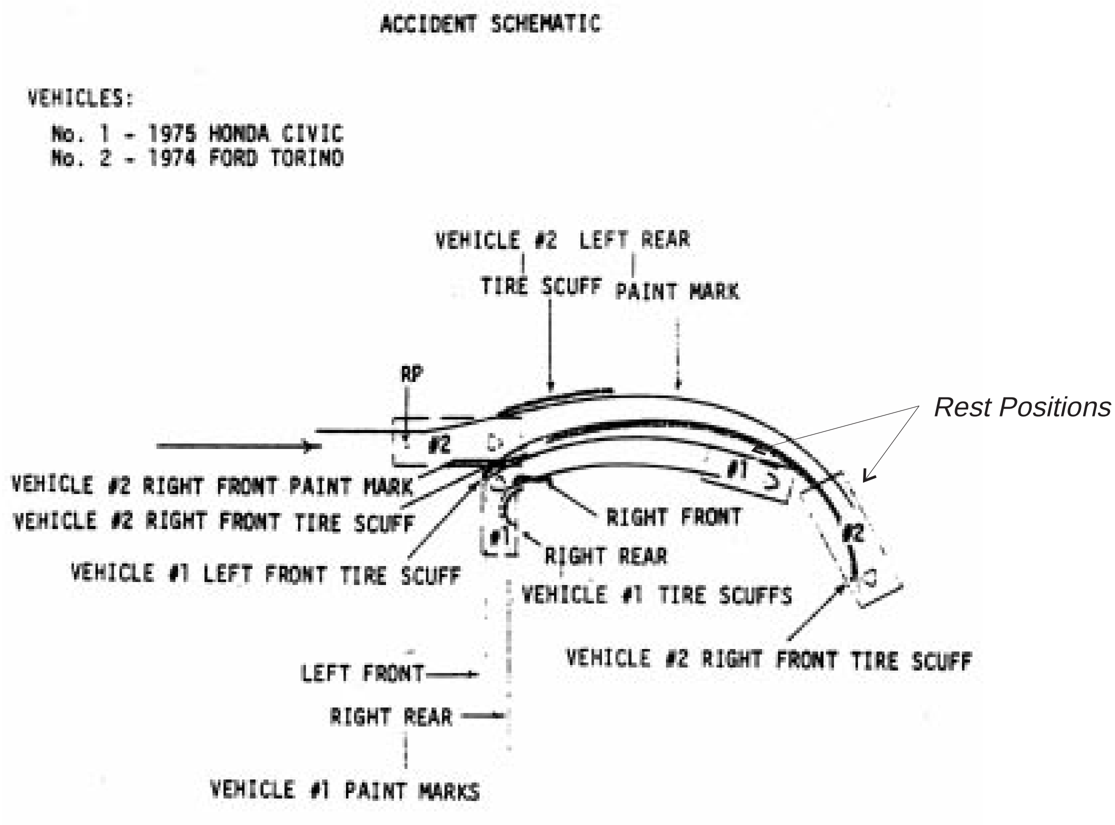
For the current paper, two of these tests, RICSAC 9 and RICSAC 12, have been selected to illustrate the validation results. Table 1 shows predicted and simulated rest positions, CDCs and delta-V's for both tests. The individual experimental results are described and discussed below.

RICSAC Test No. 9

Staged collision test no. 9 was a 90 degree collision between a Honda Civic (striking vehicle) and a Ford Torino (struck vehicle), as shown in Figure 16. The impact velocities were 21.2 mph for both vehicles. As a subject for validation, this collision test was interesting because it resulted in rapid rotation and secondary impact at the rear of the vehicles (test no. 10 may have even been more interesting as it was conducted at higher velocities, however, one of the vehicles struck a metal transformer at an unknown velocity before reaching its natural rest position; thus the test could not be used for purposes of validation).

Simulated paths are shown in Figure 17 (target positions for each vehicle at its rest position are also included to help assess the match between simulated and measured results). After impact, the Honda Civic rolled a short distance to its rest position. The Ford Torino was slightly redirected at impact by the (much lighter) Civic. The vehicle's post-impact path formed a wide circular arc, actually coming to rest to the right of its pre-impact direction of travel. Interestingly, if an investigator failed to notice and include the curved post-impact path, a simple momentum analysis would suggest the Honda Civic was backing up at the time of impact (perhaps *pulling* on the right side of the Ford Torino!). The match between the simulated and measured rest positions was excellent - less than 2 feet and 5 degrees for both vehicles. The only change required in the data set was to continue to drag at the Torino's (damaged) right front wheel (the original data set reduced the braking force at the right front wheel to 14 lb from 280 lb just after impact; the modified data maintained the braking force at 240 lb).

The impact damage was minor to both vehicles (see Figures 18 and 19). The simulated delta-Vs matched the experimental results quite well (see Table I). The secondary impact in the experiment was simulated extremely well. The CDC for the Civic was correct, except the extent was over-estimated. The simulated CDC for the Torino was also in close agreement, however, the measured damage was confined to the right front fender while the simulated damage also included a portion of the passenger compartment.



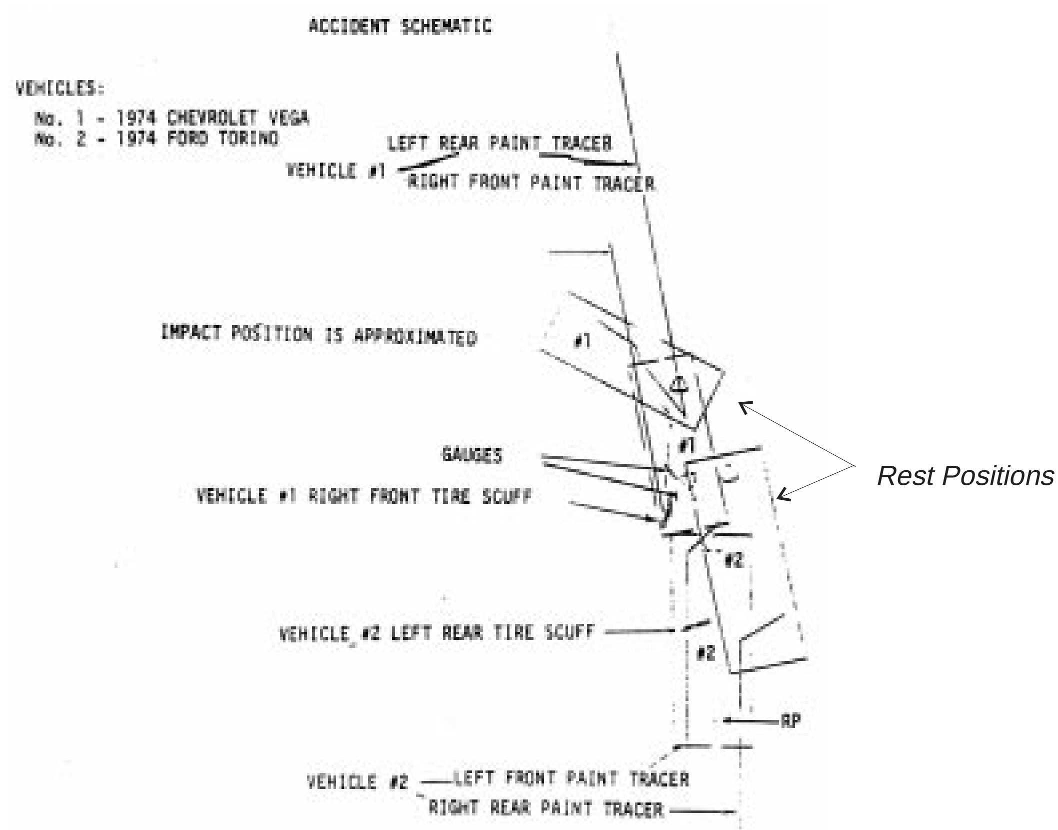


Figure 16 - RICSAC 9 (top) and RICSAC 12 (bottom) test layouts, from the original Calspan report [19].

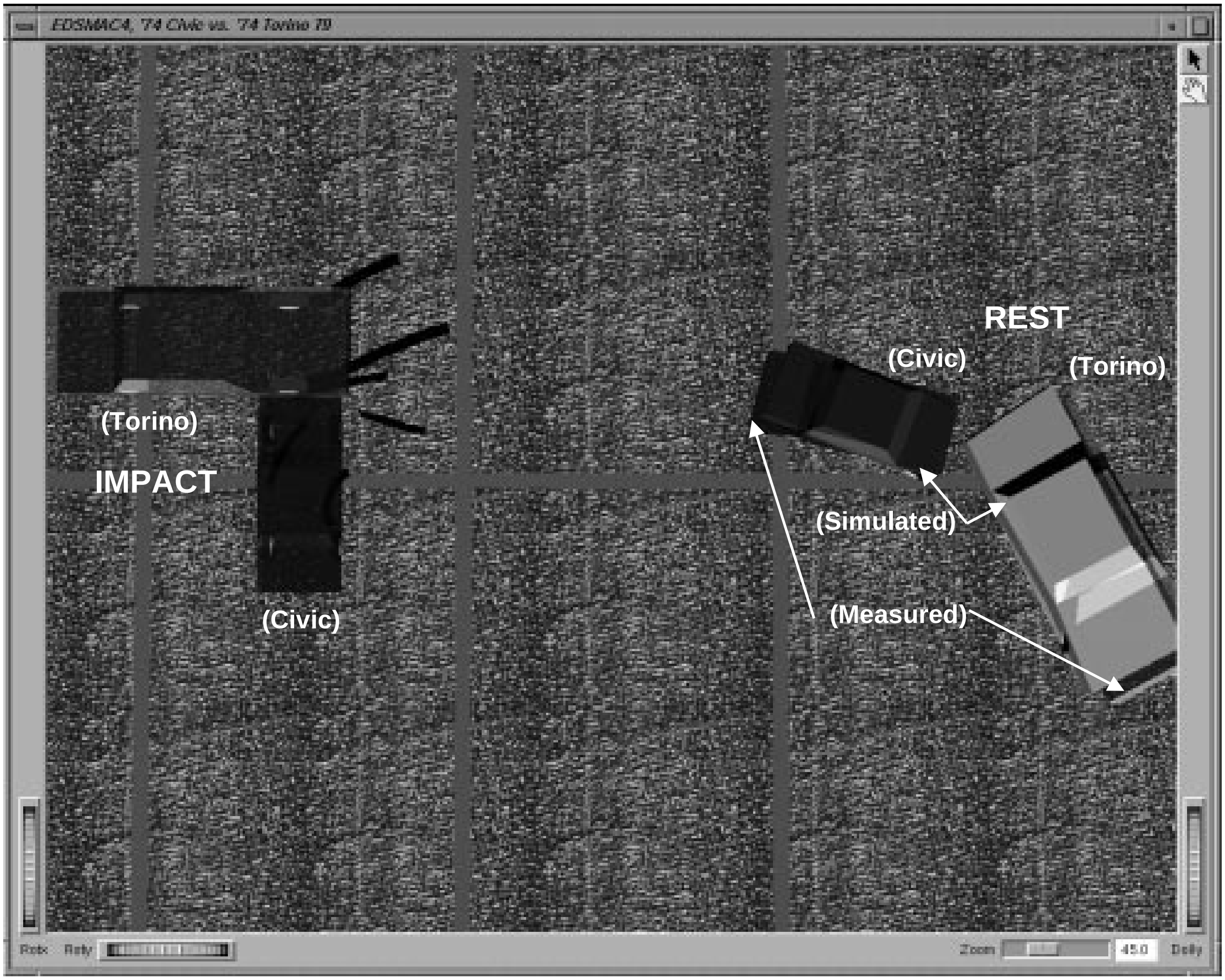


Figure 17 - RICSAC 9 results. Target vehicles are shown at measured impact and rest positions.

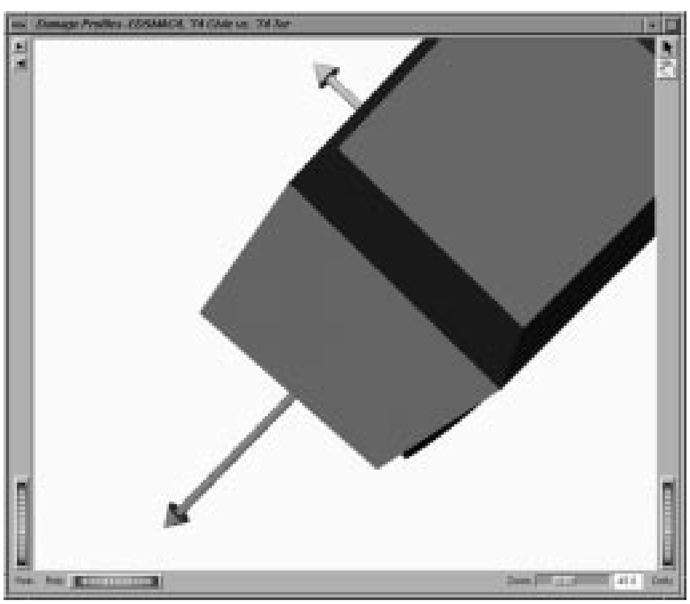


Figure 18 - RICSAC 9 Simulated damage Profile, Honda Civic

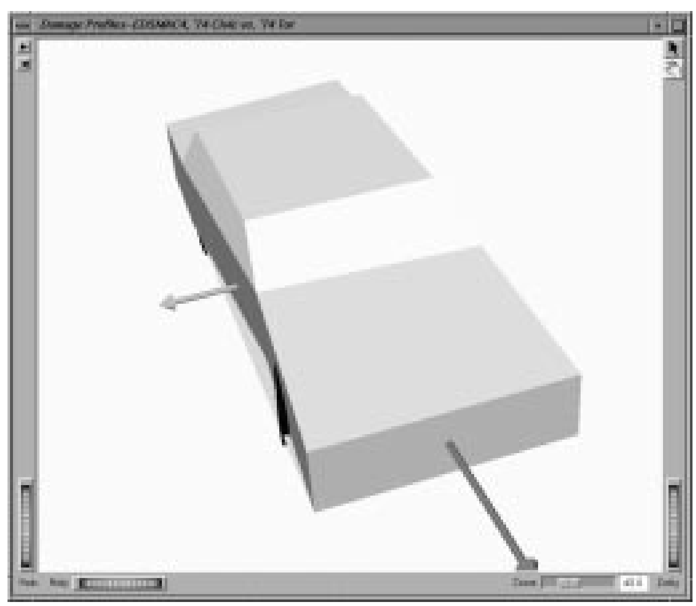


Figure 19 - RICSAC9 Simulated damage profile, Ford Torino

Table 1. RICSAC9 and RICSAC12 Test Results

Test	Method	Vehicle 1				Vehicle 2			
		V_{impact} (mph)	X,Y,Ψ (ft,ft,deg)	CDC	ΔV (mph)	V_{impact} (mph)	X,Y,Ψ (ft,ft,deg)	CDC	ΔV (mph)
Test No. 9	Measured	21.2	4.0,35.3,104.0	11FDEW2	21.4	21.2	-5.0,49.5,152.0	02RFEW2	8.9
	EDSMAC		7.7,16.8,74.1	11FDEW4	20.2		-34.0,54.9,193.7	02RYEW3	8.6
	EDSMAC4		3.8,36.1,111.5	11FDEW4	20.8		-4.3,48.1,152.9	02RYEW2	9.1
Test No. 12	Measured	31.5	22.3,-5.5,118.0	12FDEW4	40.1	31.5	6.8,2.6,-12.0	12FYEW4	26.4
	EDSMAC		22.7,-9.9,134.8	12FYEW4	40.3		8.0,1.7,-4.0	11FYEW3	27.3
	EDSMAC4		20.4,-6.4,116.7	12FDEW3	40.4		6.5,1.6,-17.9	12FYEW3	27.5

RICSAC Test No. 12

Staged collision test no. 12 was an offset head-on collision between a Chevrolet Vega and a Ford Torino, as shown in Figure 16. The impact velocities were 31.5 mph for both vehicles. This test resulted in rapid post-impact rotation of the Chevrolet Vega.

RICSAC 12 represents a relatively severe crash (60+ mph closing velocity). It is also a rather common collision configuration for countries with left-hand drive vehicles, that is, partial overlap left front-to-left front.

Simulated paths are shown in Figure 20 (target positions for each vehicle at its rest position are also included to help assess the match between simulated and measured results).

After impact, the 3150 lb Chevrolet Vega was driven backwards by the 4540 lb Ford Torino. The post-impact path lengths are rather short, less than 10 feet. The Vega rotated 55 degrees counter-clockwise and the Torino rotated 12 degrees; both vehicles rotated counter-clockwise. The Vega post-impact rotation was difficult to simulate using the original RICSAC Test 12 data set. Research revealed the original data set used a front wheel steer angle of 12 degrees, whereas the data recorder measured 51 degrees (obviously beyond the steering stops, but conceivable given the amount of frontal damage sustained by the vehicle) during the experiment. When EDSMAC4 was rerun with 50 degree front steer angles, the resulting rest orientation was correct to within 1.3 degrees.

Both vehicles sustained moderate to severe frontal crush. The measured maximum crush was 39 inches at the left front corner of the Chevrolet Vega and 40 inches at the left front corner of the Ford Torino. The damage profiles simulated by EDSMAC4 (see Figures 21 and 22) matched quite well.

DISCUSSION

Inspection of Table 1 reveals the small differences in the trajectories from EDSMAC and EDSMAC4 (typically less than two feet). However, in RICSAC9 the rest position and orientation of the Ford Torino was improved by leaving its (damaged) right front wheel partially locked (as shown in Figure 16). In RICSAC 12, the rest orientation of the Chevrolet Vega was improved by using wheel steer angles found in the recorded raw data, as described earlier. Simulated CDCs and delta-V's remain nearly unchanged.

Collision pulses (i.e., acceleration vs time histories) for RICSAC tests 9 and 12 are shown in Figures 23 and 24, respectively (only one vehicle is displayed in each graph; the history for the other vehicle is nearly identical in character, but has a lower amplitude because of its larger mass). A comparison of the pulses from EDSMAC and EDSMAC4 reveals the EDSMAC4 accelerations are greater, occur earlier, and have a shorter interval than their EDSMAC counterparts. These findings are expected and are consistent with the extensions to the force-deflection relationship in EDSMAC4: The introduction of the A coefficient produces these effects by providing a vehicle exterior with an initial stiffness characteristic. These findings should be important to those using EDSMAC or EDSMAC4 to produce a collision pulse used by an occupant simulation. Note also that comparison of the total area under the EDSMAC and EDSMAC4 curves reveals they are essentially equal. Therefore, the calculated delta-V and related trajectory kinematics are not greatly affected by the change in the force-deflection relationship (as observed in the earlier discussion on trajectories).

Using the EDSMAC4 collision algorithm to visualize vehicle damage profiles sometimes results in aberrations. For example, the headlight in the damage region might look like it is stretched along the side of the vehicle rather than being

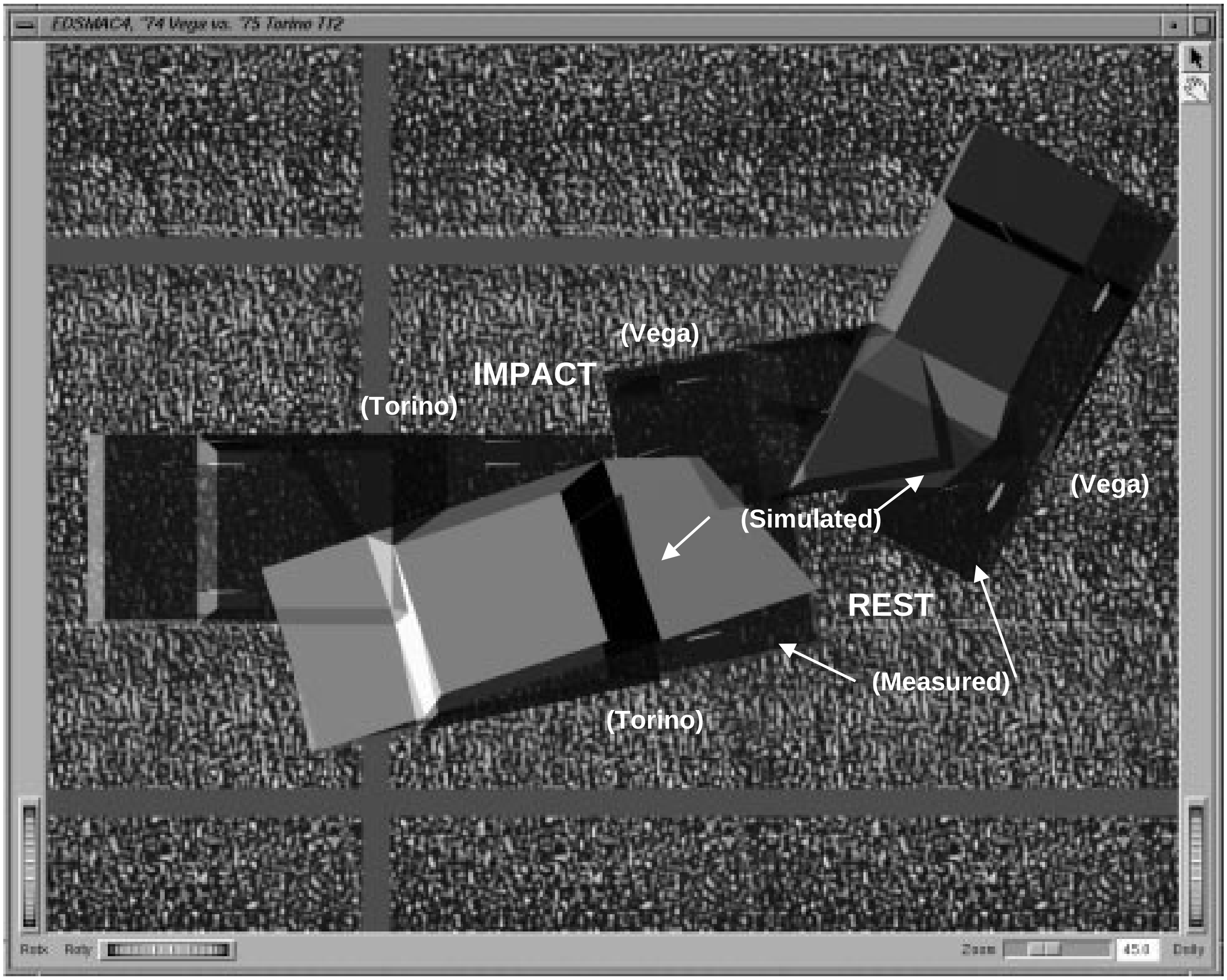


Figure 20 - RICSAC 12 results. Target vehicles are shown at measured impact and rest positions.

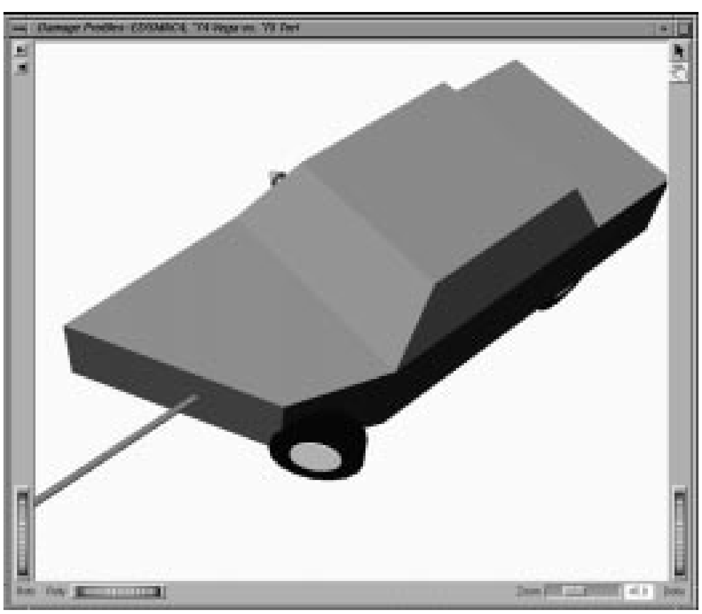


Figure 21 - RICSAC 12 Simulated damage Profile, Chevrolet Vega

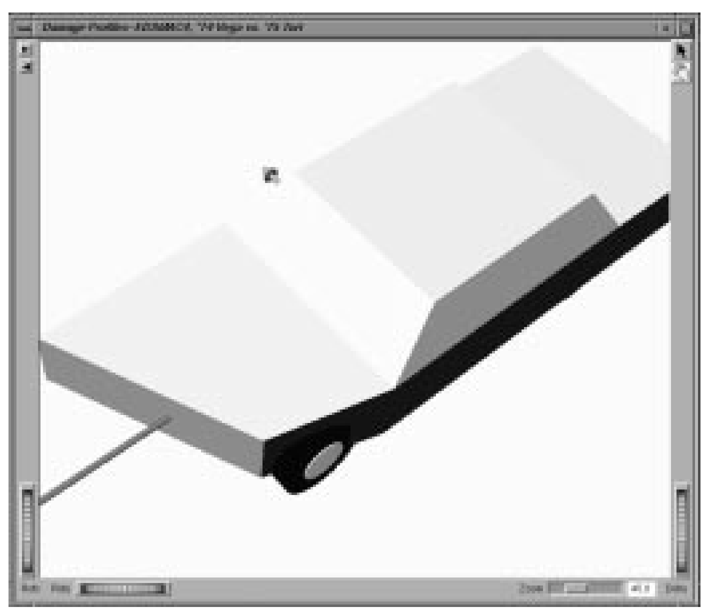


Figure 22 - RICSAC12 Simulated damage profile, Ford Torino

pushed in the thrust direction. This occurs because of the way the damage algorithm works: vertices are always moved in the direction of the RHO vectors, which may be different from the direction the vertex actually moved during the crash. This visual artifact does not invalidate the physical damage profile results.

The reader is referred to previous papers [1,2]. The current paper builds on the work in those papers, and thus, the reader can better understand the similarities and differences between EDSMAC and EDSMAC4 by reviewing those papers in conjunction with the current work. In particular, nothing in this work has made EDSMAC invalid, rather, it has been extended with new capabilities.

The current paper does not include a validation of collisions involving articulated vehicles. Unfortunately, no well-instrumented collision experiments involving tractor-trailers have been found in the literature, thus, comparisons between simulation and experiment are not possible. However, it should be noted that the models incorporated into EDSMAC4 have been used in previous models with success (albeit not in the same model). That is, the model for articulation forces and moments comes from the well-validated Phase 4/EDVDS model [17] and the model for collision forces and moments between articulated vehicles is the same as for individual vehicles, except that the new model must also handle the condition wherein no force is calculated on RHO vectors unless the vehicle has jackknifed (as explained earlier; see Figure 13).

The wheel displacement algorithm in EDSMAC4 was not used in any tests included in this report. However, experiments conducted internally have shown a single wheel displacement may have a significant effect on a vehicle's simulated rest position. Insufficient experience has been gathered using the blow-out model to draw any useful conclusions.

CONCLUSIONS

1. A comparison of simulation results from EDSMAC and EDSMAC4 revealed small differences in the simulated rest positions, delta-Vs and CDCs for the RICSAC9 and RICSAC12 staged collision experiments.

2. Some differences were attributed to changes in the RICSAC input data to better reflect the actual test data.

3. Extensions to the model were also responsible for the differences between EDSMAC and EDSMAC4 results. The primary difference in collision-phase results was due to the extended force-deflection model. The difference in post-collision trajectories was due to the addition of load transfers and, secondarily, small differences in departure conditions associated with the new collision model.

4. The changes to the force-deflection model had the expected effect on the collision phase results: The deformations were reduced, the acceleration peaks were higher and occurred earlier, and the duration of the collision phase was shorter.

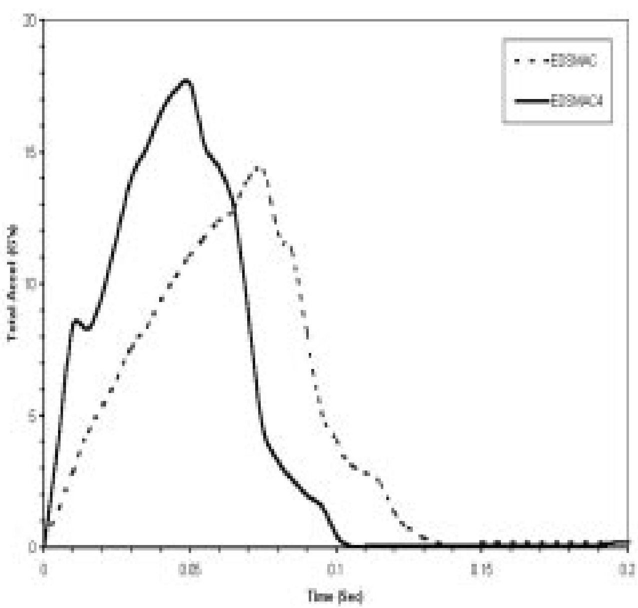


Figure 23 - RICSAC 9 Acceleration Pulse, EDSMAC and EDSMAC4 results

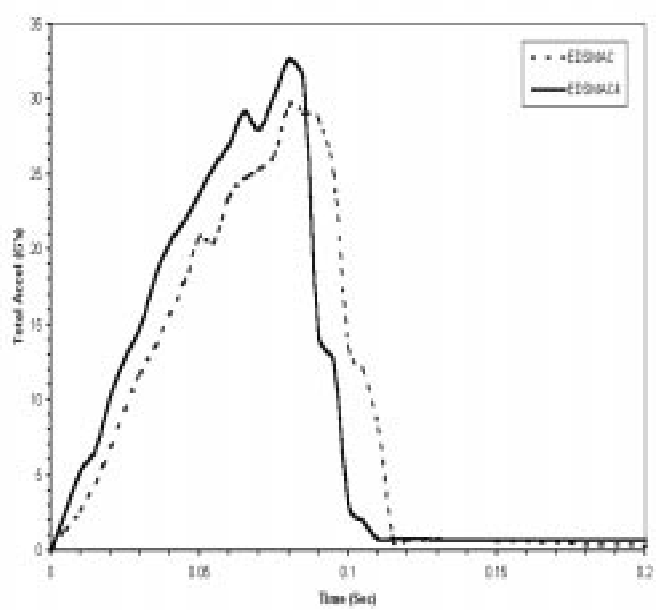


Figure 24 - RICSAC 12 Acceleration Pulse, EDSMAC and EDSMAC4 results

Also, as expected, little effect was observed in the post-impact trajectories.

5. Damage visualization served an important purpose. The procedure used, `MakeProfile()`, was simple and straightforward. However, some interpretation of the 3-D damage profiles may be required because the direction of deformation is always towards the vehicle CG; this may not be the case during the actual crushing. Also, the profile was produced using a 2-dimensional algorithm; the algorithm did not change the z (i.e., elevation) damage coordinates.

6. The ability to study articulated vehicles and vehicles having tandem axles and dual tires was a useful extension to the model.

7. Wheel displacements, if known, should be considered the analysis.

RECOMMENDATIONS

1. A comprehensive study of barrier and pole impacts is recommended. Such a study would help to validate the model as well as provide useful information regarding the analysis of these events.

2. A comparison of results using EDSMAC4, EDVTS and EDVDS for articulated vehicle maneuvers is recommended. Such as study would help to validate the EDSMAC4 articulated vehicle model.

3. A series of articulated vehicle collision simulations should be performed. Although it may not be possible to compare these results with staged collision experiments, such tests would serve to confirm the reasonableness of the results. These simulation tests should include a mix of vehicle types (i.e., cars, small trailers and on-highway trucks with single, double and triple trailers).

4. Further research should be performed regarding the role of post-impact tire deflation and wheel displacement on the resulting vehicle path.

REFERENCES

1. 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.

2. Day, T.D., Hargens, R.L., "Further Validation of EDSMAC Using the RICSAC Staged Collisions," SAE Paper No. 900102, Society of Automotive Engineers, Warrendale PA, 1990.

3. Day, T.D., "Validation of Several Reconstruction and Simulation Programs in the HVE Simulation Environment," SAE Paper No. 960891, Society of Automotive Engineers, Warrendale PA, 1996.

4. Woolley, R.L., Kinney, J.R., "Reference Cases for Comparison of Collision Algorithms Used in Accident Reconstruction," SAE Paper No. 940567, Society of Automotive Engineers, Warrendale, PA, 1994.

5. Fonda A.G., "Energy and Major Diversion in Accident Reconstruction," SAE Paper No. 960888, Society of Automotive Engineers, Warrendale, PA, 1996.

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

7. McHenry, R. "Development of a Computer Program to Aid the Investigation of Highway Accidents," Calspan Report No. VJ-2979-V-1, DOT HS-800 821, December, 1971.

8. 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.

9. *HVE User's Manual, Version 2*, Engineering Dynamics Corporation, Beaverton, OR 1998.

10. Day, T.D., "A Computer Graphics Interface Specification for Studying Humans, Vehicles and the Environment," SAE Paper No. 930903, presented at the SAE International Congress and Exposition, Detroit, MI, February, 1993.

11. Day, T.D., "An Overview of the HVE Vehicle Model," SAE Paper No. 950308, presented at the SAE International Congress and Exposition, Detroit, MI, February, 1995.

12. Day, T.D., "An Overview of the HVE Developer's Toolkit," SAE Paper No. 950308, presented at the SAE International Congress and Exposition, Detroit, MI, February, 1995.

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

14. Collision Deformation Classification, SAE Recommended Practice J-224BMAR80, Society of Automotive Engineers, Warrendale, PA, 1980.

15. Campbell, K.L., “An Energy Basis for Collision Severity,” SAE Paper No. 74565, Society of Automotive Engineers, Warrendale, PA, 1974.

16. Day, T.D., Hargens, R.L., “Application and Misapplication of Computer Programs for Accident Reconstruction,” SAE Paper No. 890738, Society of Automotive Engineers, Warrendale, PA 1988.

17. Day, T., “Differences Between EDVDS and Phase4,” SAE Paper No. 1999-01-0103, Society of Automotive Engineers, Warrendale, PA 1999.

18. Blythe, W., Day, T., Grimes, W., “3-Dimensional Simulation of Vehicle Response to Blow-outs,” SAE Paper No. 980221, presented at the SAE International Congress and Exposition, Detroit, MI, February, 1998.

19. Jones, I., Baum, A., “Research Input for Computer Simulation of Automobile Collisions - Volumes I - IV, DOT HS-805-040, December, 1978.

20. HVE Case Files SMA4Validation1-4, SMA4Validation5-8, SMA4Validation9-12, Engineering Dynamics Corporation, Beaverton, OR 1998.