3D Contact Algorithm for Tire-Road Interaction

Mauri Vesimäki
MBS Models Oy, Finland

1. Introduction

The 3D contact algorithm for tire-road interaction was originally developed with offroad applications in mind, but has since proven to be of interest in variety of other areas of applications as well. There had long existed tire models for handling on a flat road and for durability testing on a 2D-road. The main motivation for this work was lack of an easy-to-use tire model, which could combine both of these domains allowing engineers to look into, for example, stability of a forest machine, while cornering on a deep slope, even or uneven. The main objectives for such a tire model were:

- enable cornering on an uneven 3D-road surface,
- road definition should be based on geometry,
- allow varying friction based on position on the road,
- allow road to move under a vehicle,
- take the tire width into account, and
- allow arbitrary tire carcass shape definition.

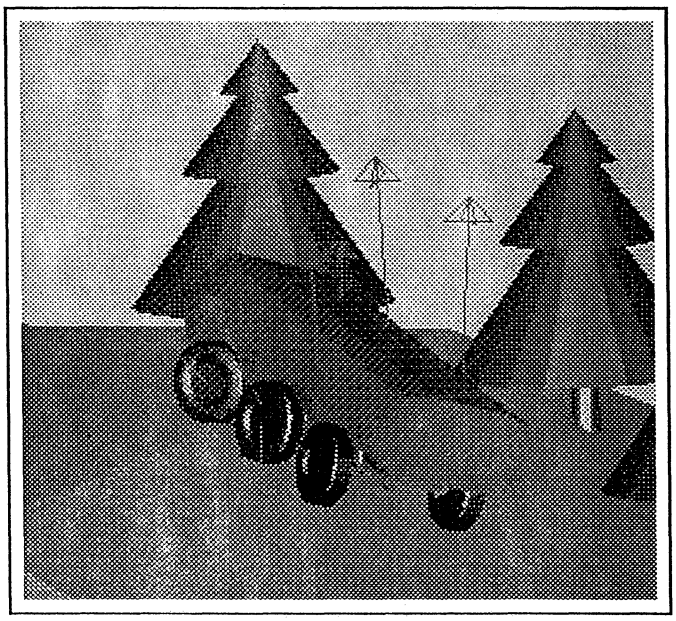

Figure 1: Offroad testing of armored wheeled vehicle XA-185.
Model and analysis courtesy of Patria Vehicles Oy.

Because there are no discontinuities in the physical world of tire-road interaction and because existence of those in a mathematical model always creates serious numerical problems, the algorithm was carefully developed to be continuous in all cases. This also contributes to the speed of simulations - the smoother and more continuous the model is the faster ADAMS/ Solver typically executes.

Even though the current version of the algorithm is rather generic it does not take all physical phenomena into account. It is known to carry limitations especially related to handling of

small, sharp obstacles. The algorithm is under development in order to more realistically handle even the small obstacles and for optimization of speed of simulations.

The 3D contact algorithm is used to compute effective normal, contact point etc. of the road and, thus, can relatively easily be coupled with any force computation routines (i.e. lateral force due to slip etc.). The algorithm uses standard ADAMS/Tire road definition file format and a version compatible with ADAMS/Tire force computations is also available. In the examples shown in this paper there were an in-house, Pacejka 87 based, force computation routine applied.

2. Concept of Moving Road

Implementation of the applied force computation method defines how the contact algorithm interacts with your ADAMS model. The in-house Pacejka 87 relies on standard GFORCEs. The road definition requires only a road reference marker for each road. A road and its reference can be shared by multiple tires in a model.

The road reference marker may belong to the any part in your model. The road does not move with respect to this part, but the part itself may move. This is useful for example if you want to drive your vehicle onto a train, a weight, or anything which is suspended itself. You may also attach another road to your vehicle's chassis to find out if the tire hits the chassis during a driving maneuver or not.

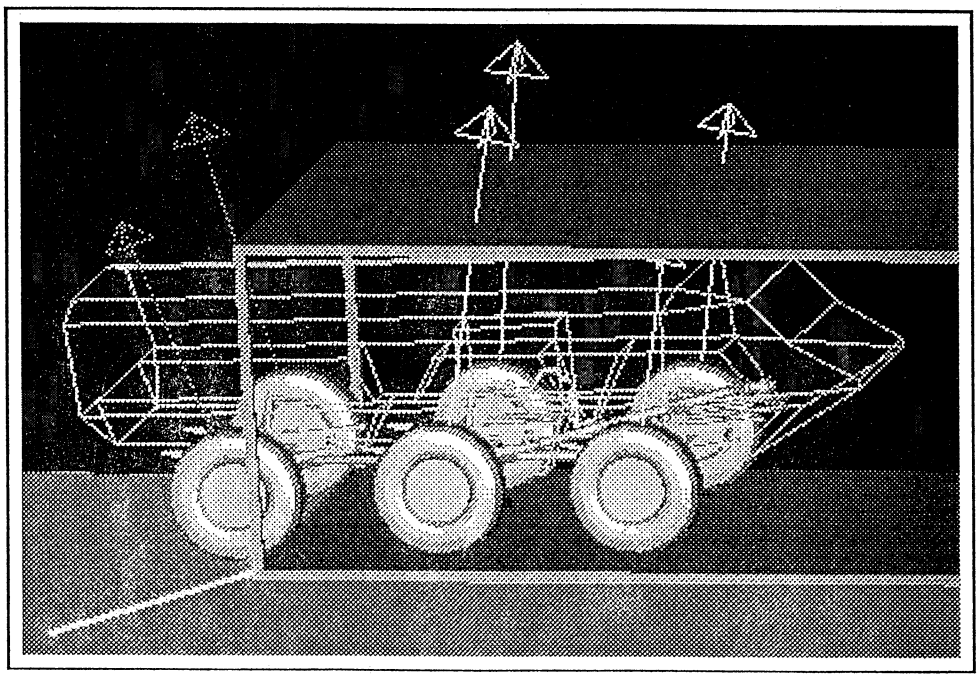

Figure 2: Armored wheeled vehicle XA-185 driving onto a suspended platform. Model and analysis courtesy of Patria Vehicles Oy.

All road nodes (coordinates) are given in the road reference coordinate system. If you visualize your road using a shell graphics in ADAMS/View with the same reference marker, you get WYSIWYG appearance of the road for your animations.

NOTE: One tire can interact with one road only. If you wish to have several roads interacting with the same physical tire, simply create several copies of the tire in your model, each with their own road.

3. 3D Contact Geometry Computations

Given the location and the carcass shape of the tire and the road geometry the contact algorithm resolves

a) effective penetrated volume of a tire,
b) effective penetration of a tire,
c) effective road contact point location,
d) effective road normal at effective road contact point, and
e) effective road friction coefficient at the contact point.

The contact algorithm takes the following steps:

a) interpolate tire carcass shape to given number of equally spaced points,
b) divide tire into cross sections with equal widths,
c) build-up a volume-penetration look-up table,
d) read in road data file and initialize road memory tables,
e) find road elements potentially in contact with tire,
f) find road elements, which are in contact with tire,
g) eliminate such portions of road elements, which are behind other elements,
h) compute effective penetrated volume,
i) compute effective road contact point location,
j) compute effective road normal at road contact point,
k) compute effective road friction coefficient at contact point, and
l) use look-up table to resolve effective penetration of a tire.

Road geometry is defined as a set of two-sided triangular elements, which may be in any position and angle with respect to each other. Number of road elements is virtually unlimited.

The contact algorithm is completely general in nature, so it is capable of handling any number of road elements simultaneously in contact with tire. In the current version, though, number of such elements is limited to 100. It is important to realize that each contact between a road element and a tire is treated independently as a continuous (in longitudinal direction) contact, not as a point contact. Therefore number of simultaneous contact points is always infinite, though number of elements is limited. Using continuous contact instead of finite number of discrete contact points makes the algorithm more accurate and guarantees numerical continuity in all cases.

3.1. Interpolation of Tire Carcass Shape

Tire carcass shape can be defined as a set of points. If shape is not given then a rectangular shape based on radius and width of tire is used. Carcass shape is assumed symmetric over the center line of a tire, and therefore shape points needs to be entered only for a half width of the tire. Given tire carcass shape is interpolated linearly to given number of equally spaced points (see Figure 3).

Carcass shape is defined in terms of relative values (scales). Absolute coordinate values for the shape are computed by multiplying relative values with tire unloaded radius and half width respectively. Tire relative width must be given in an ascending order from 0.0 through 1.0, where value 0.0 corresponds to the center line of the tire.

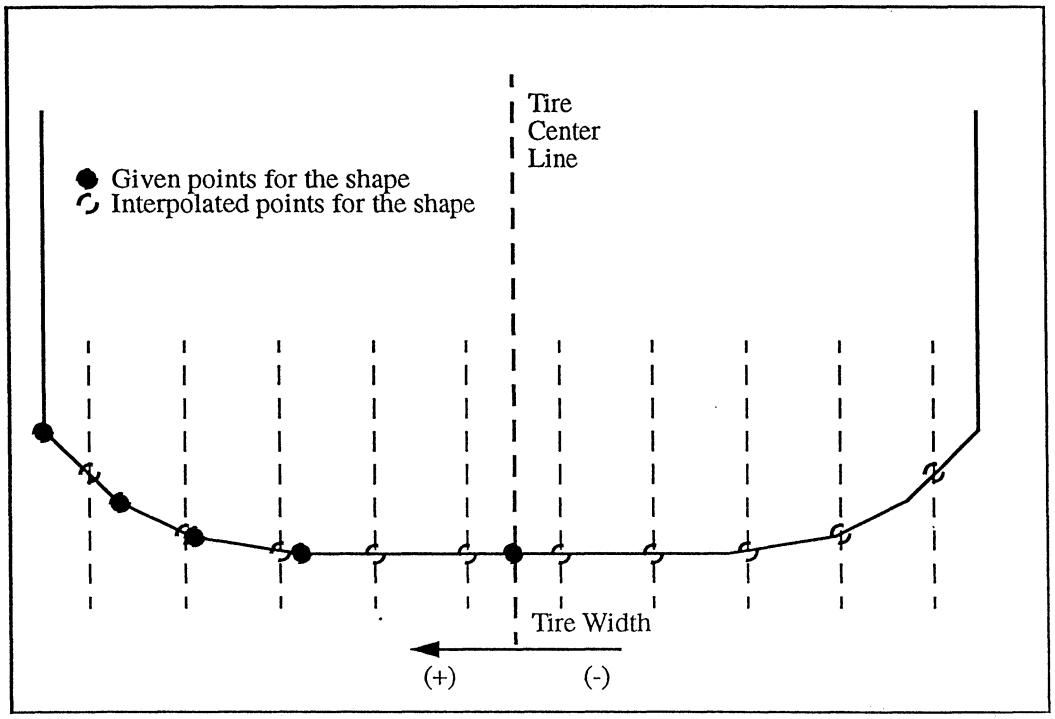

Figure 3: Definition of the tire carcass shape, given shape and interpolated values

3.2. Divide Tire Into Cross Sections with Equal Widths

Tire is then divided in lateral cross sections (see Figure 4) passing through the interpolated points of the tire carcass shape. Contact computations are performed only on these lateral cross sections assuming that the geometrical road contact occurring on these cross sections represent an average of the contact along the width of the corresponding discrete pieces of tire carcass shape. An obstacle, which is narrower in tire lateral direction than the distance between neighboring cross sections may pass undetected.

When a larger number of interpolated points are used then the geometrical solution obviously becomes more accurate, but it also consumes more CPU.

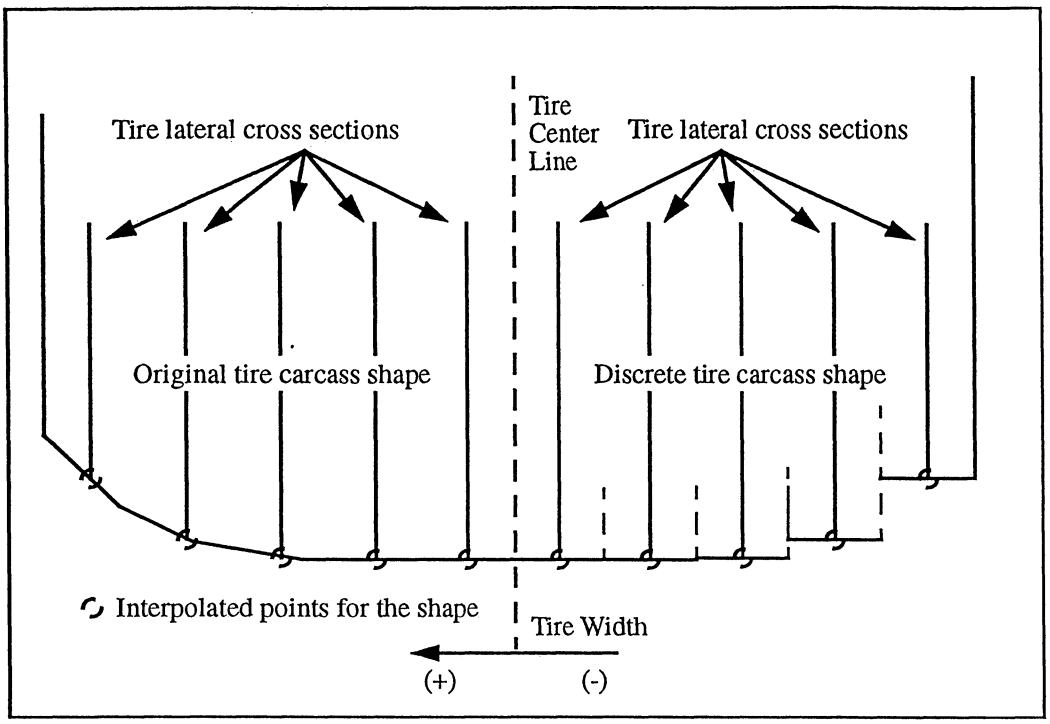

Figure 4: Tire lateral cross sections

3.3. Build-Up a Volume-Penetration Look-Up Table

A look-up table is being computed internally in order to relate effective penetrated volume to effective penetration of tire. Effective volume is computed for a set of penetration values assuming that the semidiscrete tire carcass shape were penetrated vertically into a flat surface.

3.4. Read in Road Data File

Road data file will be read in preprocessed and converted to SI-units. All computations are performed internally in SI-units, and therefore algorithm should behave numerically in the same way despite of set of model units used.

For increased efficiency the contact algorithm precomputes a set of memory tables for each road during initialization. In addition, it reads road data into the memory only once per each road, i.e. if two tires share the same road, then that road will be read in only once in order to save both memory and CPU. Though, it is allowed to use independent roads for each tire.

3.5. Find Road Elements Potentially in Contact with Tire

Attempting to gain more speed the algorithm eliminates some of the road elements, before actual contact computations. Only the elements, which are regarded as potential candidates for contact are further studied.

3.6. Find Road Elements, Which Are in Contact with Tire

The second pass scans through all potential contact elements one by one and checks for a contact against each tire lateral cross section independently. After that the algorithm knows all locations, where road elements go through tire cross sections. If we consider any cross section and road element intersection alone, it is a line on the plane of the cross section. Together all intersection lines form one or many polylines (with "normal" roads, there is usually only one polyline per cross section), which represent the shape of the road under this particular cross section of the tire (see Figure 5)

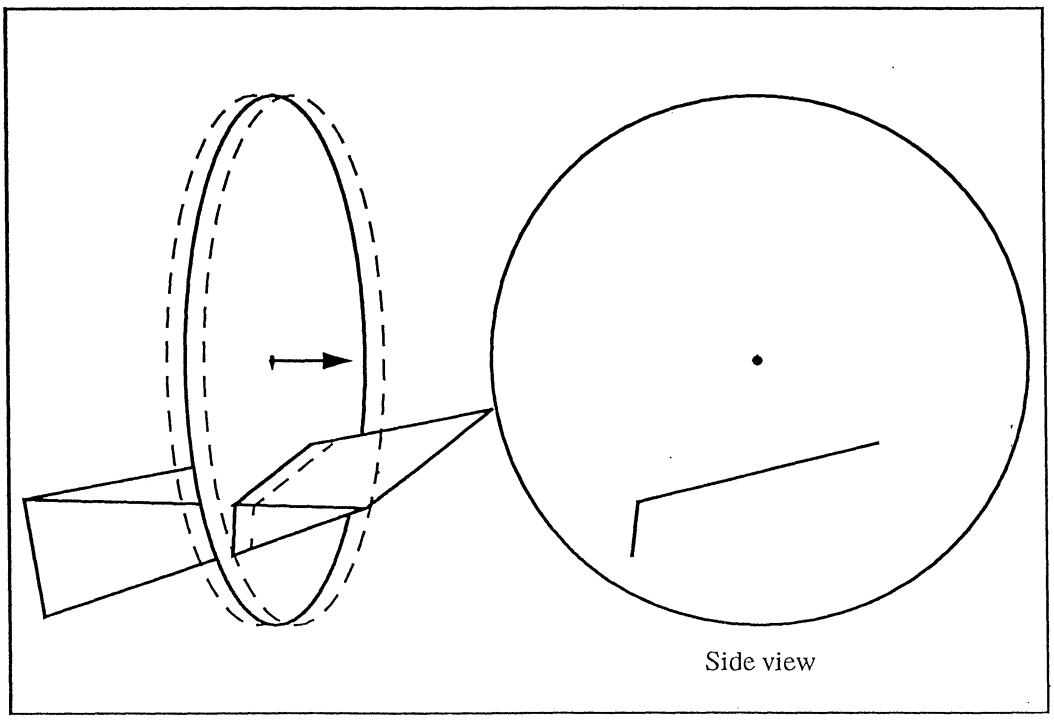

Figure 5: Intersecting road elements and a tire cross section

In Figure 5 the dashed circles beside tire lateral cross section visualize the width of the cross section. In the same figure road consist of only two road elements for clarity. Their intersection lines with the tire cross section were intentionally left short. For the contact algorithm road is a finite piece of geometry. Tire may run off the road without limitations. This is particularly useful when modeling for example vehicle from one surface to another, for example driving a vehicle on a traincar.

3.7. Eliminate Shadowed Portions of Road Elements

If a road element is shadowed by an other element, partially or in whole, it must not be regarded, while computing contact volume. For example, if for some reason there are two road elements on top of each other with a small or zero offset in the road normal direction, a tire will penetrate through both of those and "sense" twice its stiffness, if both of the road elements are taken into account.

Therefore the contact algorithm eliminates shadowed portions away from intersection polylines before proceeding. Elimination routine allows only those portions of intersection polylines, which are directly "visible" from the tire cross section center (see Figure 6).

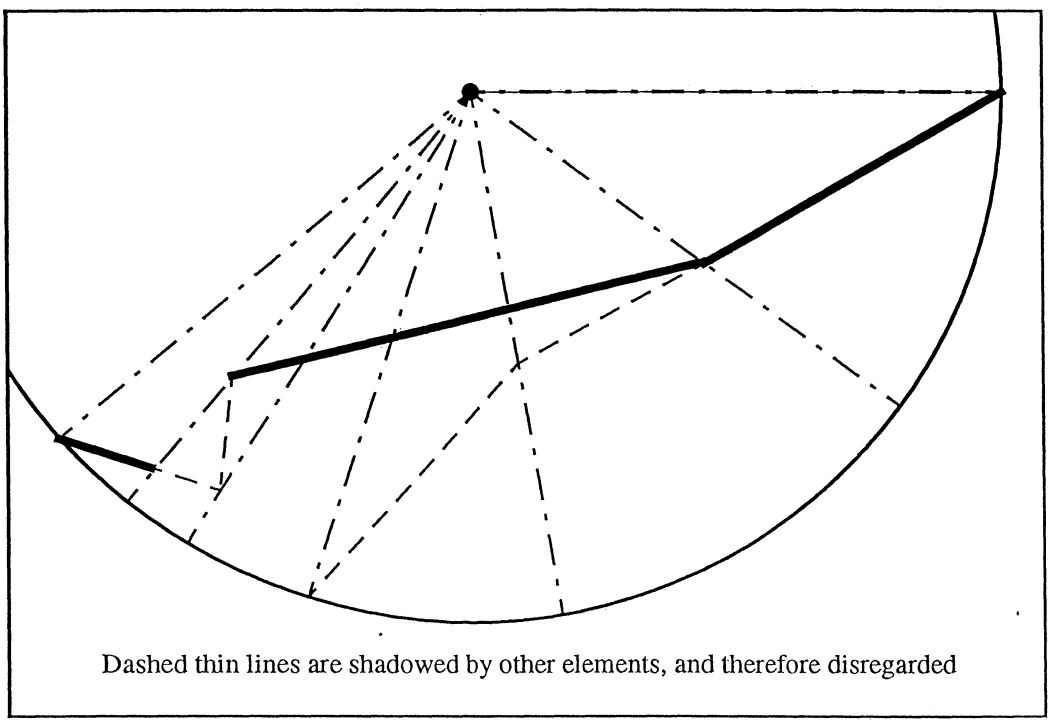

Figure 6: Eliminating shadowed road elements

3.8. Compute Effective Penetrated Volume

Based on the road surface polyline computed earlier, the penetrated area of a tire cross section can now divided in discrete pieces (see Figure 7). Each of those pieces share a common width, the width of the tire cross section, and therefore penetrated area for that cross section can be derived simply by multiplying areas of the penetrated segments with their width.

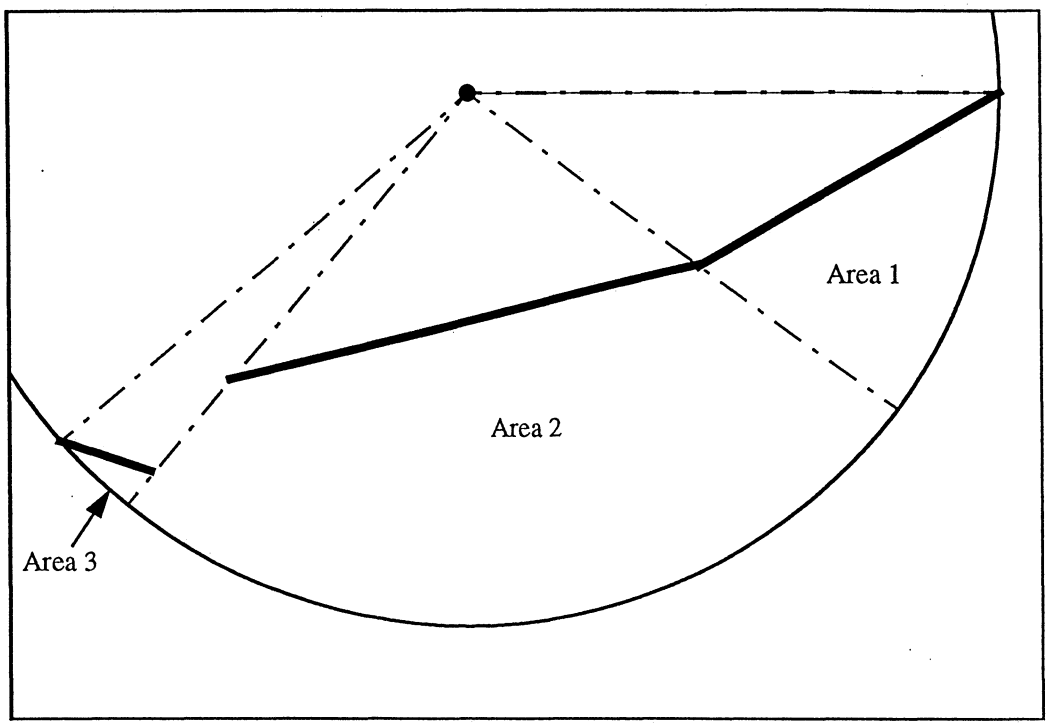

Figure 7: Penetrated areas of a tire cross section

Penetrated volumes are first summed up for each cross section then the total volumes of cross sections are summed together.

$$V_n = A_n * W_{cross\_section} \quad \textbf{(Eq. 1)}$$

where V_n = penetrated volume of a segment of a cross section
A_n = area a segment of a tire cross section
$W_{cross\_section}$ = width of a cross section

$$V_{eff} = \Sigma_{cross\_sections} \Sigma_{segments} V_n \quad \textbf{(Eq. 2)}$$

where V_{eff} = effective penetrated volume of a tire
$\Sigma_{cross\_sections}$ = sum over all cross sections
$\Sigma_{segments}$ = sum over penetrated segments of a cross section

3.9. Compute Effective Road Contact Point Location

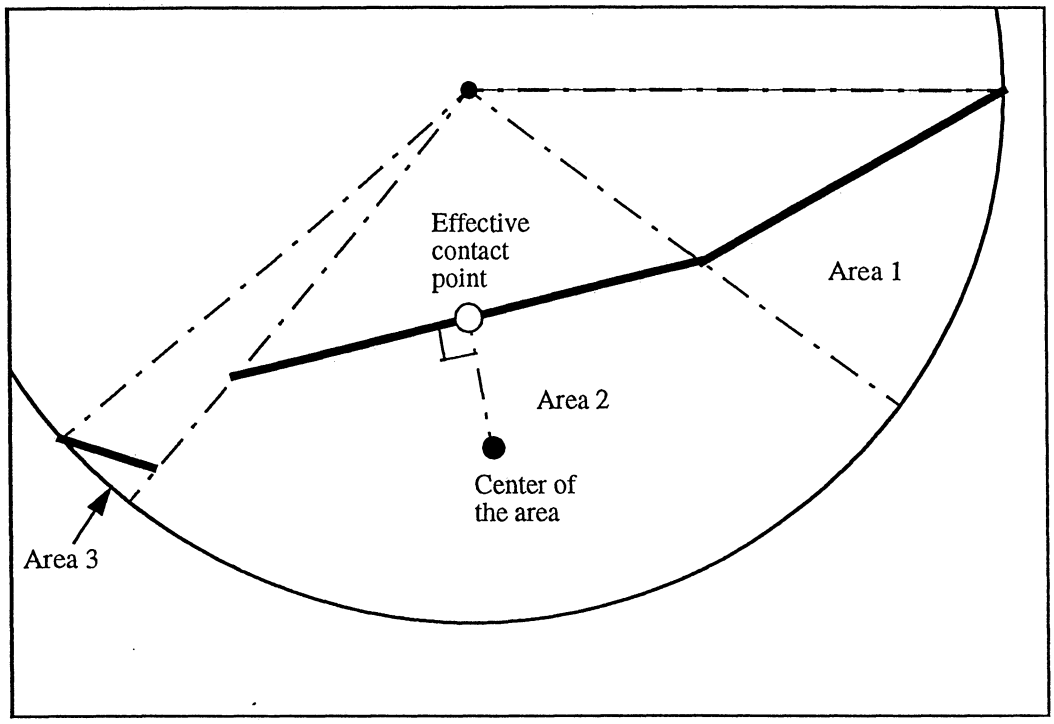

Figure 8: Effective contact points of penetrated areas

In order to find the effective road contact point, the algorithm first computes contact point coordinates for each piece of penetrated volume, and then takes their weighted average as the effective point of contact. Contact point for each piece of volume is found by drawing a line perpendicular to the road surface through the center of the area, and then resolving coordinates of the intersection of that line and the road surface (see Figure 8).

Coordinates for the effective contact point for the tire are computed as follows:

$$X_{ecp} = \Sigma_{cross\_sections} \Sigma_{segments} V_n / V_{eff} * X_{cpn} \quad \text{(Eq. 3)}$$

where

X_{ecp}	= x-coordinate of the effective contact point
$\Sigma_{cross\_sections}$	= sum over all cross sections
$\Sigma_{segments}$	= sum over penetrated segments of a cross section
V_n	= penetrated volume of nth segment
V_{eff}	= effective penetrated volume of the tire
X_{cpn}	= x-coordinate of the contact point of nth segment

Y- and Z-coordinates of the effective contact point are computed analogical to X.

3.10. Compute Effective Road Normal at Road Contact Point

Effective road normal is derived in a similar manner that the effective road contact point. Road normal vectors of each piece of penetrated volume are weighted with the value of their penetrated volume and then summed up to the effective road normal.

$$X_{ern} = \Sigma_{cross\_sections} \Sigma_{segments} V_n/V_{eff} * X_{rnn} \quad \text{(Eq. 4)}$$

where

X_{ern}	= x-component of the effective road normal
$\Sigma_{cross\_sections}$	= sum over all cross sections
$\Sigma_{segments}$	= sum over penetrated segments of a cross section
V_n	= penetrated volume of nth segment
V_{eff}	= effective penetrated volume of the tire
X_{rnn}	= x-component of the road normal corresponding to the nth segment

Y- and Z-components of the effective road normal are computed analogical to X.

3.11. Compute Effective Road Friction Coefficient

Once more, the same weighting algorithm is applied to road element friction coefficients to derive effective road friction coefficient at the contact point location. Road friction coefficients of each piece of penetrated volume are weighted with the value of their penetrated volume and then summed up to the effective road friction coefficient.

$$u_e = \Sigma_{cross\_sections} \Sigma_{segments} V_n/V_{eff} * u_n \quad \text{(Eq. 5)}$$

where

u_e	= effective road friction coefficient
$\Sigma_{cross\_sections}$	= sum over all cross sections
$\Sigma_{segments}$	= sum over penetrated segments of a cross section
V_n	= penetrated volume of nth segment
V_{eff}	= effective penetrated volume of the tire
X_{rnn}	= road friction coefficient corresponding to the nth segment

3.12. Use Look-Up Table to Resolve Effective Penetration of a Tire

During initialization there were a volume-penetration look-up table built-up. Effective penetration of a tire is resolved from effective penetrated volume by applying linear interpolation on the look-up table.