Title
Low Speed Collision Dynamics: Second Year Report

Authors
Tongue, Benson
Moon, Ahrie
Harriman, Doug

Publication Date
1996

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Low Speed Collision Dynamics: Second Year Report

Benson Tongue
Ahrie Moon
Doug Harriman

California PATH Research Report
UCB-ITS-PRR-96-8

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

April 1996

ISSN 10551425

Low Speed Collision Dynamics: Second Year Report

Benson H. Tongue, Ahrie Moon, and Doug Harriman
Department of Mechanical Engineering
University of California at Berkeley

California PATH Program

Abstract

The primary aim of this project is to develop a simulation program which will accurately determine platoon dynamics during both nominal and emergency situations on the road. An important objective was to construct a program that is both user friendly and highly modular. By properly designing this program, one can quickly switch between controllers and platoon models, allowing easy evaluation of design changes. The overall simulation program includes individual modules that supply control input force, aerodynamic drag force, and road-tire interaction forces. There also are modular subroutines which contain the model for the engine dynamics and the transmission. The structure of the computer simulation code allows for a choice of vehicle type and dynamic complexity.

Keywords: Collision Dynamics, Intelligent Vehicle Highway Systems (IVHS), Safety, Traffic Platooning

Nomenclature

A_f	frontal area of the car
β	vehicle sideslip angle
C_D	coefficient of aerodynamic resistance
δ_k	steering angle input to the k-th tire
Δ_i	spacing between vehicles i and $i-1$ in a platoon
$\overline{\Delta_i}$	A; normalized by the average car length
$EF(n)$	correction factor obtained from extrapolation curves
F_{aero}	aerodynamic drag force
F_{A_k}	sum of the forces on the k-th tire in the vehicle's longitudinal direction
F_{B_k}	sum of the forces on the k-th tire in the vehicle's lateral direction
$F_{collision}$	collision force (a function of x_{crush})
F_{damp_k}	damping force of the suspension joint over the k-th tire
F_o	break out force (used in collision force calculation)
F_{roll}	tire rolling resistance force
F_{spring_k}	spring force of the suspension joint over the k-th tire
g	gravitational acceleration
h_2	vertical distance from the vehicle's center of gravity to the roll center
h_4	vertical distance from the vehicle's center of gravity to the pitch center
h_5	longitudinal distance from the vehicle's center of gravity to the pitch center
I_x	vehicle body moment of inertia about x-axis
I_y	vehicle body moment of inertia about y-axis
I_z	vehicle body moment of inertia about z-axis
k_{body}	linear spring constant representing the stiffness of the vehicle body
l_1	longitudinal distance from the vehicle's center of gravity to the front axle
l_2	longitudinal distance from the vehicle's center of gravity to the rear axle

m_{EF_0}	slope of the extrapolation line corresponding to $\overline{\Delta_i} = 0$
m_{EF_1}	slope of the extrapolation line corresponding to $\overline{\Delta_i}$ = 1
m_i	mass of vehicle i
M_x	sum of the moments on the vehicle about the z-axis
M_Y	sum of the moments on the vehicle about the y-axis
M_z	sum of the moments on the vehicle about the z-axis
n	number of cars in the platoon
ϕ_i	roll angle of vehicle i
ϕ_i	time rate of ϕ_i of vehicle i
ϕ_i	time rate of $\dot{\phi}_i$ of vehicle i
ψ_i	yaw angle of vehicle i
$\dot{\psi}_i$	time rate of ψ_i of vehicle i
$\ddot{\psi}_i$	time rate of $\dot{\psi}_i$ of vehicle i
ρ_{air}	density of air
ρ_{road}	radius of curvature of the road
R_{C_D}	reduction factor for C_D
$R_{C_D}(n)$	reduction factor for C_D in an n-vehicle platoon
s_{b1}	track of the front axle
s_{b2}	track of the rear axle
θ_i	pitch angle of vehicle i
$\dot{\theta}_i$	time rate of θ_i of vehicle i
$\ddot{\theta}_i$	time rate of $\dot{\theta}_i$ of vehicle i
v_{x_i}	longitudinal velocity of vehicle i
v_{y_i}	lateral velocity of vehicle i
v_{z_i}	vertical velocity of vehicle i
x_i	longitudinal position of vehicle i
x_b	active bumper length
x_{crush}	amount of crush of the body frame

X	inertial coordinate axis fixed to the road, orthogonal to Y-axis
$\vec{x}_i$	state vector for the body of vehicle i
$\dot{\vec{x}}_i$	time derivative of the state vector for the body of vehicle i
y_i	lateral position of vehicle i
Y_S	lateral deviation from the road centerline
Y	inertial coordinate axis fixed to the road, orthogonal to X-axis
z_i	vertical position of vehicle i
χ	vehicle velocity angle with respect to the inertial coordinate system

1 Executive Summary

The goal of this project is to develop a user-friendly simulation program which will allow the study of platoon dynamics in both nominal and emergency scenarios. This report discusses the issues investigated during the second phase of the project.

There is relatively little literature on modeling the type of multibody system that is represented by the platoon paradigm used in the PATH project, i.e. a system consisting of multiple bodies which are not physically connected. When contact occurs during collisions, the dynamic interactions are complex. The existing literature on multibodies is intended for applications such as robotics and satellites where the interacting bodies are connected at all times, and does not fully address the dynamic responses that can occur in a platoon.

In platooning research to date, simple vehicle models have been utilized in order to examine platoon dynamics. When used in longitudinal control development, the vehicles are essentially modeled as point masses. When a collision occurs within the platoon during longitudinal operation, it is assumed that the vehicles remain in a straight line ([Hedrick et al. 1991] [Sheikholeslam 1989]). In a eral control law research, the vehicle models are slightly more sophisticated in that they include the possibility of lateral translation and yawing motions [Peng et al. 1991]. The models developed so far, however, do not account for the complex interactions between vehicles during a collision. To realistically model platoon dynamics during a wide range of possible scenarios, it is necessary to investigate these complex interactions between vehicles. This is the aim of the current work.

During the first phase of the project, a simple vehicle model was developed for preliminary analysis, and a modular simulation program was written which utilized this simple vehicle model. Some of the preliminary issues were collision detection and numerical integration code modification to account for collisions within the simulation. Also, the ground work was laid out for a modular simulation program.

In the second phase of the project, the simple vehicle model developed for preliminary analysis was upgraded to include an engine model, transmission model, a combined lateral

and longitudinal controller, and a more detailed collision dynamics model. Also, the models for the vehicle dynamics and aerodynamics have been upgraded. The simulation program now allows different vehicle types within the platoon and a choice of dynamic complexity in the vehicle models. Section 2 describes the additions to the vehicle model and the control input model. Section 3 discusses the collision dynamics model. Section 4 documents the structure of the modular simulation program.

2 System Models

Initially, a simple vehicle model was developed to be utilized in testing a modular collision simulation program. Some of the issues involved in collision detection and post-collision dynamics were addressed using this model.

After some preliminary analysis, improvements have been made to the vehicle model, collision dynamics model, and aerodynamic drag model. The vehicle model now includes an engine model, a transmission model, and a suspension model. For the collision dynamics model, the vehicle body is no longer modeled as a rigid rectangle, but includes distinct and nonlinear crush characteristics that serve to model the vehicle's structural deformation during collisions. The aerodynamic drag model has been refined to include the effect of other vehicles on the air flow over a given vehicle in a platoon [Zabat et al. 1995]. The tire model is a Bakker-Pacejka model which utilizes a curve fit of Yokohama tire test data [Peng 1992]. The controller implemented for the simulations is the combined longitudinal-lateral controller developed in [Pham 1995].

2.1 Vehicle Dynamics Model

The simple rectangular vehicle model has been extended so that the vehicle contains a rigid passenger compartment, front and rear crush zones and bumpers. The sprung mass of the

vehicle body has six degrees of freedom, translation in three orthogonal directions (x, y, z) and rotation about all three axes. The vehicle model now includes an engine model and a transmission model developed in [McMahon1994].

2.1.1 Vehicle Body

The complete set of the equations of motion for the lateral and longitudinal dynamics are derived from earlier work in [Peng1992]. Refer to figure 1 for the coordinate system description of the vehicle. The positive s-direction coincides with the forward direction of the vehicle. The states of the vehicle's sprung mass are positions and velocities in translation and rotation. These are described in Table 1.)

Table 1: Vehicle States

State	Description
x	longitudinal position
y	lateral position
z	vertical position
v_x	longitudinal velocity
v_y	lateral velocity
v_z	vertical velocity
ψ	yaw
$\dot{\psi}$	yaw rate
θ	pitch
$\dot{\theta}$	pitch rate
ϕ	roll
$\dot{\phi}$	roll rate

There are five other states associated with the mass center of the car. These are the inertial coordinate in X-axis, inertial coordinate in Y-axis, lateral deviation from the road center (y_s), rate of y_s, and the road heading angle. These states arise because of the need

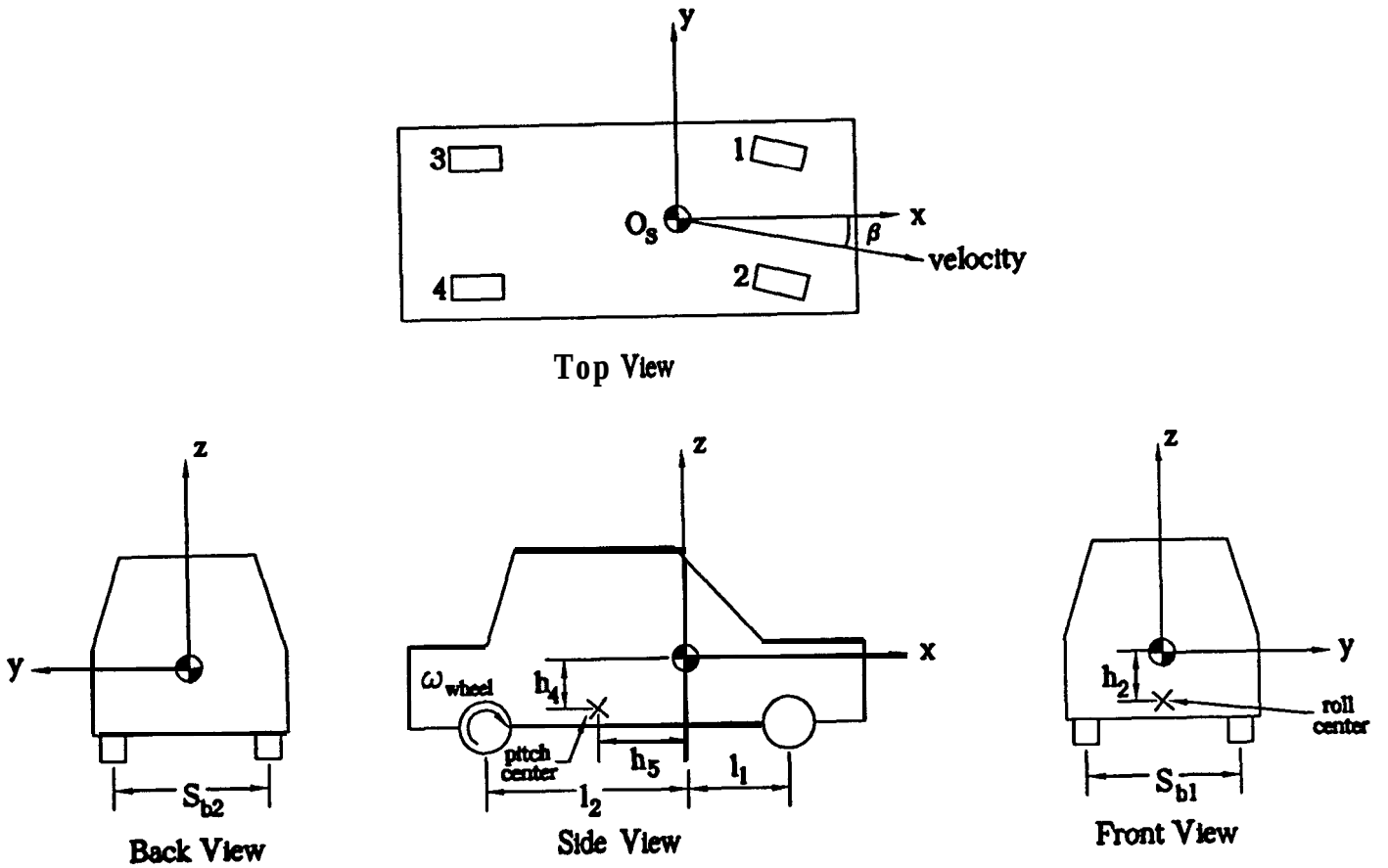

Figure 1: Vehicle Coordinate System

to keep track of the car's position with respect to the road. Note that the X-Y coordinate system is an inertial coordinate system fixed to the road at some point; however, the x-y coordinate system is fixed to the mass center of the vehicle.

The following are the equations of motion for the vehicle body, assuming small $\psi, \theta,$ and ϕ. The subscript i which indicates the number of the vehicle in the platoon has been suppressed.

$$m[\dot{v}_x - v_y\dot{\psi} + h_4\ddot{\theta} + h_2\dot{\phi}\dot{\psi} + h_2\phi\ddot{\psi}] = \sum_{k=1}^{4} F_{A_k} - F_{aero_x} F_{c_x} - F_{roll} \tag{1}$$

$$m[\dot{v}_y + v_x\dot{\psi} - h_2\ddot{\phi} + h_4\dot{\theta}\dot{\psi} + h_4\theta\ddot{\psi}] = \sum_{k=1}^{4} F_{B_k} - F_{aero_y} \cdot I F_{c_y} \tag{2}$$

$$m[\dot{v}_z + v_z\dot{\chi}\beta - h_5\ddot{\theta}] = \sum_{k=1}^{4} F_{P_k} - mg \tag{3}$$

$$I_x[\ddot{\phi} - \theta\ddot{\psi} - \dot{\theta}\dot{\psi}] - (I_y - I_z)\dot{\theta}\dot{\psi} = M_x - \theta M_z \tag{4}$$

$$I_y[\ddot{\theta} + \phi\ddot{\psi} + \dot{\phi}\dot{\psi}] - (I_z - I_x)\dot{\phi}\dot{\psi} = M_y + \phi M_z \tag{5}$$

$$I_z[\ddot{\psi} + \theta\ddot{\phi} - \ddot{\theta}\phi] - (I_x - I_y)\dot{\theta}\dot{\phi} = M_z + \theta M_x - \phi M_y \tag{6}$$

where m is the mass of the vehicle,

h_2 is the vertical distance from the vehicle's center of gravity to the roll center,

h_4 is the vertical distance from the vehicle's center of gravity to the pitch center,

h_5 is the longitudinal distance from the vehicle's center of gravity to the pitch center,

F_{aero} is the aerodynamic force,

$F_{collision}$ is the collision force,

F_{roll} is the force due to the rolling resistance of the tires,

χ is the vehicle velocity angle with respect to the inertial reference frame,

β	is the vehicle side slip angle,
I_x	is the vehicle moment of inertia about the x-axis,
I_y	is the vehicle moment of inertia about the y-axis,
I_z	is the vehicle moment of inertia about the z-axis,
M_x	is the sum of the moments on the vehicle about the x-axis,
M_y	is the sum of the moments on the vehicle about the y-axis, and
M_z	is the sum of the moments on the vehicle about the z-axis.

F_{A_k} is the sum of the forces on the ki-th tire in the vehicle's longitudinal direction, i.e. the vehicle's x-direction. The x_k direction corresponds to the k-th tire's longitudinal axis, and the y_k direction corresponds to the k-th tire's lateral axis. (See figure 2.)

$$F_{A_k} = F_{x_k} \cos \delta_k - F_{y_k} \sin \delta_k \quad k = 1, \ldots, 4 \tag{7}$$

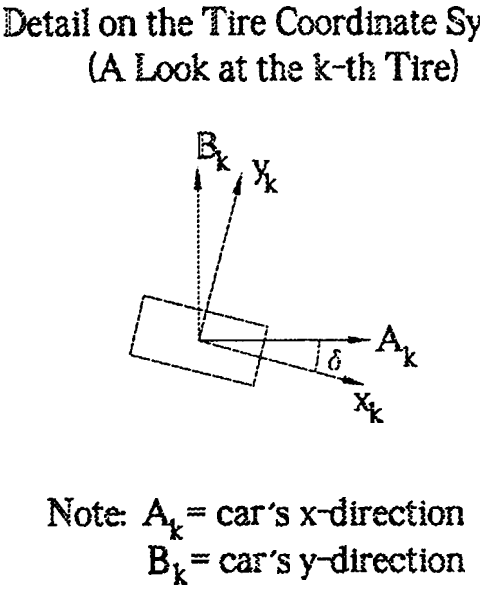

Figure 2: Tire Coordinate System

F_{B_k} is the sum of the forces on the k-th tire in the vehicle's lateral direction, i.e. the vehicle's y-direction.

$$F_{B_k} = F_{x_k} \sin \delta_k + F_{y_k} \cos \delta_k \qquad k = 1, \ldots, 4 \tag{8}$$

F_{P_k} is the sum of the forces on the k-th tire in the vertical direction and includes forces from the suspension.

$$F_{P_k} = F_{spring_k} + F_{damp_k} \qquad k = 1, \ldots, 4 \tag{9}$$

where F_{spring_k} is the spring force of the suspension joint over the k-th tire, and
F_{damp_k} is the damping force of the suspension joint over the k-th tire.

The following are the equations for the moments on the sprung mass due to the forces on the tires.

$$\begin{aligned} M_{x_{tire}} = {} & (\frac{s_{b1}}{2} + h_2\phi)F_{P_1} + (\frac{s_{b2}}{2} \text{ t } h_2\phi)F_{P_3} - (\frac{s_{b1}}{2} - h_2\phi)F_{P_2} \\ & -(\frac{s_{b2}}{2} - h_2\phi)F_{P_4} - (z - h_5\theta)\sum_{k=1}^{4} F_{B_k} \end{aligned} \tag{10}$$

$$M_{y_{tire}} = (l_2 + h_4\theta)(F_{P_3} + F_{P_4}) - (l_1 - h_4\theta)(F_{P_1} + F_{P_2}) - (z - h_5\theta)\sum_{k=1}^{4} F_{A_k} \tag{11}$$

$$\begin{aligned} M_{z_{tire}} = {} & (l_2 - h_4\theta)(F_{B_1} \text{ t } F_{B_2}) - (l_1 - h_4\theta)(F_{B_3} \text{ t } F_{B_4}) - (\frac{s_{b1}}{2} \text{ t } h_2\phi)F_{A_1} \\ & -(\frac{s_{b2}}{2} \text{ t } h_2\phi)F_{A_3} \text{ t } (\frac{s_{b1}}{2} - h_2\phi)F_{A_2} \text{ t } (\frac{s_{b2}}{2} - h_2\phi)F_{A_4} \end{aligned} \tag{12}$$

where s_{b1} is the track of the front axle,
s_{b2} is the track of the rear axle,
l_1 is the longitudinal distance from the vehicle's center of gravity to the front axle, and
l_2 is the longitudinal distance from the vehicle's center of gravity to the rear axle. rear axle.

2.1.2 Engine

The two-state engine model developed in [McMahon1994] is used here. The states of the engine are the manifold air mass and the engine speed. Inputs to the engine are the controller

specified throttle angle and the pump torque from the torque converter. The actuator dynamics between the controller and the throttle valve is approximated as a first-order system.

2.1.3 Transmission

The drive train model from [McMahon1994] has been slightly modified for use here. The model in [McMahon 1994) includes a torque converter model, actuator dynamics for the steering input, and a dummy gear-shifting model. (The dummy gear-shifting model stays in third gear at all times.) This gear-shifting model has been replaced by one which shifts gears depending on the engine speed. (Appendix A) Inputs to the torque converter are the engine speed, the average angular velocity of the wheels, and the engaged gear ratio. The outputs are the pump torque and the turbine torque. The turbine torque is used to obtain the shaft torque.

2.2 Aerodynamic Drag Model

The aerodynamic drag model is a modified version of the standard drag model for a car traveling on a straight road. The modifications correct for the effect of other cars in the platoon on the air flow over a given vehicle in the platoon.

The starting point for the aerodynamic model is the basic aerodynamic model for a car traveling on a straight road:

$$F_{aero} = (1/2)\rho C_D A_f u_x^2 \tag{13}$$

where F_{aero} is the aerodynamic force,
ρ is the density of air (standard value = 0.00238 kg/m^3),
C_D is the coefficient of aerodynamic resistance,
A_f is the frontal area of the car,
v_x is the vehicle speed in the direction along the longitudinal axis of the car.

This force is assumed to act in the direction opposite to the vehicle's velocity.

In the present work the frontal area of the car, A_f, has been changed to the "effective area" perpendicular to the velocity vector, and $\dot{u}_i$ has been changed to the magnitude of the velocity vector. (See figure 3.) This modification is made to account for when the car velocity vector is not perpendicular to the frontal plane of the vehicle.

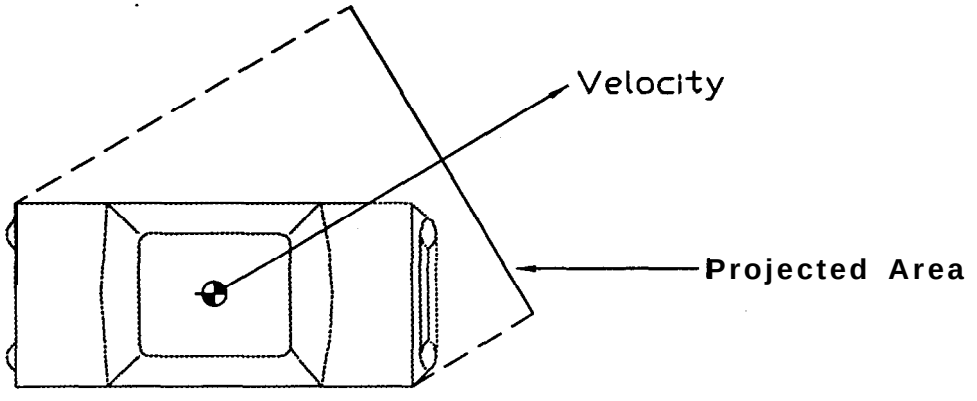

Figure 3: Aerodynamic Model Using "Effective Area"

The second modification to this basic aerodynamic drag model is the variability of C_D due to the presence of other vehicles in a platoon. In a platoon where the spacing between cars is small (usually less than one or two car lengths), the aerodynamic drag is reduced for all vehicles. This reduction is represented by R_{C_D}, which is defined as the ratio of C_D of a vehicle in the platoon to C_D of the vehicle in isolation. R_{C_D} is a function of $\overline{\Delta_i}$ and the number of vehicles in a platoon, where $\overline{\Delta_i}$ is Δ_i divided by the average car length of the platoon. The present model has been derived from the experimental data for R_{C_D} in [Zabat et al. 1995]. The results from four-vehicle platoons have been used to determine R_{C_D} as a function of $\overline{\Delta_i}$. R_{C_D} for vehicles in larger platoons have been obtained by extrapolation.

The relationship between R_{C_D} and $\overline{\Delta_i}$ depends on the position of the vehicle in the platoon. A simple curve fit of the lead vehicle data for a four-vehicle platoon gives the following relationship between R_{C_D} and $\overline{\Delta_i}$:

for $\overline{\Delta_i} <= 0.3$

$$R_{C_D} = 0.6 + (\overline{\Delta_i}/6) \tag{14}$$

for $\overline{\Delta_i} > 0.3$

$$R_{C_D} = 1 - 0.35 \exp(-\lambda(\overline{\Delta_i} - 0.3)) \tag{15}$$

$$\lambda = \ln 7/0.7 \tag{16}$$

For the two interior vehicles, the experimental data are similar. Therefore, only one curve was fit to the data and the resulting R_{C_D} vs. $\overline{\Delta_i}$ relationship is given by

$$R_{C_D} = 0.46 + 0.28\overline{\Delta_i}. \tag{17}$$

For values of $\overline{\Delta_i}$ which give $R_{C_D} > 1$, R_{C_D} is set equal to 1 for the interior vehicles. Similarly, for the last vehicle the R_{C_D} to $\overline{\Delta_i}$ relationship is given by

$$R_{C_D} = 0.6 + 0.12\overline{\Delta_i}. \tag{18}$$

Again, for values of $\overline{\Delta_i}$ which give this $R_{C_D} > 1$, R_{C_D} is set equal to 1 for the last vehicle.

For platoons containing more than four vehicles, R_{C_D}'s from the above relationships are modified using extrapolation of experimental data with two, three, and four-vehicle platoons. This modification is done using the following algorithm:

1. Determine whether the vehicle is the lead vehicle, a interior vehicle, or the last vehicle. Using the current value of $\overline{\Delta_i}$, calculate a preliminary R_{C_D} for that vehicle from the four-vehicle platoon equations.

2. Find the slope of the line in the modified extrapolation curve plot corresponding to the value of $\overline{\Delta_i}$, This is done using the equation

$$m_{EF}(\overline{\Delta_i}) = m_{EF_0} - (m_{EF_0} - m_{EF_1})\overline{\Delta_i}. \tag{19}$$

where

- m_{EF_0} slope of the line corresponding to $\overline{\Delta_i}$ = 0, which is 0.6533
- m_{EF_1} slope of the line corresponding to $\overline{\Delta_i}$ = 1, which is 0.2933.

If $m_{EF}(\Delta_i)$ is negative, set $m_{EF}(\overline{\Delta_i}) = 0$.

3. Calculate $EF(n)$ using the correction factor from the extrapolation line equation corresponding to an n-vehicle platoon

$$EF(n) = 1 - m_{EF}(\overline{\Delta_i}) * (1 - 1/n). \tag{20}$$

4. Calculate $R_{C_D}(n)$, the reduction factor for C_D of a vehicle in an n-vehicle platoon, using the equation

$$R_{C_D}(n) = (EF(n)/EF(4)) * R_{C_D}. \tag{21}$$

2.3 Tire Model

The tire model is a Bakker-Pacejka model [Peng 1992] which utilizes a curve fit to Yokohama tire test data.

Both the Bakker-Pacejka model and the previously used model produced qualitatively similar behavior. Since the Bakker-Pacejka model is based on more recent data, it was decided to switch over to it.

The Bakker-Pacejka tire mechanics model allows for a combination of lateral and longitudinal slip. The inputs to the tires are the steering angle, traction forces, and braking forces. The details for the tire model can be found in [Peng 1992].

Several assumptions have been made to simplify the code. All tires are assumed to be identical, and so they all have the same empirical coefficients in the model. (If desired, it is possible to treat each tire individually in the present framework of the simulation code.) Also, the forces on the tires from the road are assumed to be directly transferred to the vehicle via the point where the tires are attached to the vehicle.

2.4 Control Input Model

The program has been designed such that the function that calculates the control input is modular in form. This feature allows the program to be utilized as a test bed for various

controllers. The program can accept control inputs during emergency as well as nominal operating conditions.

The controller implemented for the simulations is the combined longitudinal-lateral controller developed in [Pham 1995]. Control effort is maintained during and after a collision.

3 Collision Dynamics

In the first phase of the project, the vehicles were considered to be rigid and the interactions between the vehicles during collisions were modeled by impulse-momentum formulations. In the present work, the collision dynamics model allows the vehicles to undergo controlled deformations with front and rear crush zones. These crush zones are modeled by nonlinear springs whose properties have been extracted from National Highway Transportation Safety Administration (NHTSA) frontal crash data.

3.1 Crush Zone Modeling

In order to accurately model the dynamic behavior of a platoon during non-nominal operation, a model of the physical interaction between cars is needed. Such a model requires data about the dissipation and absorption of energy of automobile frames during a collision. The model should capture both the effects on the dimensions of the carriage of the vehicle, as well as changes to the position, velocity, and acceleration of the center of mass due to collision forces. This model should have a basis in experimental data and must produce an efficient collision simulation code.

To solve the collision dynamics problem three modeling techniques were considered. The first solution is to use finite element modeling. While this technique can provide detailed deformation data, it requires extensive information about the vehicle, and is relatively slow

in converging to a solution. The next solution was to use an off the shelf collision modeling package. While these models can provide quite accurate collision simulations they can be prohibitive in both cost and computation time, as well as being difficult to interface with the rest of the platoon simulation package. The solution that was chosen was to develop a simplified model that would provide acceptably accurate energy absorption information as well as simple vehicle deformation information.

3.1.1 NHTSA Database

The first step in the development of the model was to select a database of vehicle information that would provide enough physical data to construct a dynamic model of reasonable accuracy. Two types of databases were considered, both dealing with actual vehicle collisions. The first type is built of up actual accident records. The most prevalent of these, the National Accident Sampling System contains information on thousands of accidents collected over the span of many years. While this database contains a large amount of general information about vehicle deformation and accident survivability, it is not suitable for modeling purposes due to the lack of pre-collision information. The second type of database considered consists of compilations of multiple crash tests. While these databases abound, only the NHTSA Reset and Development Vehicle Crash Test database provided information on a variety of different vehicles with both front and rear collision.

The NHTSA Database contains information on approximately 2100 controlled vehicle collisions staged by NHTSA and independent contractors. The tests were performed on a variety of vehicles in a variety of configurations from direct head-on wall collisions to two vehicle offset collisions. The vehicles represent most domestic and many foreign auto makers, and encompass model years 1971 to the present.

For each test, which is defined as a collision of any configuration, there are approximately 225 data records. These records are divided into five major categories, although all categories do not apply for all tests. The categories are: Test Configuration, Vehicle Data, Barrier Data, Occupant Data (test dummies), and Instrumentation Package data. For the collision

dynamics model, the total data set was filtered so as to leave only vehicle data for cars of model years 1985 to the present which had been tested in both frontal and rear barrier collisions with direct impacts. After this process there remained data sets for 18 vehicles. From these data sets, a majority of the data records were removed, leaving only information pertaining to the vehicles physical attributes and the test data.

3.1.2 Body Model

The reduced data set provided enough information to construct crush energy parameter plots as described in [Strother et al. 1987]. The crush energy parameter plots are used to develop relationships between the crush energy parameter of the collision (a function of the initial kinetic energy of the vehicle) and the post collision deflection of the vehicle body. Although vehicles will rebound some from the collision, the crash tests did not report rebound velocity. Since this velocity is usually quite low, its associated kinetic energy level is small compared to that of the initial velocity of the vehicle and has been neglected in the calculation of crush energy. Once an energy-crush relationship has been developed, a force-crush relationship can be found via differentiation. This relationship is of the form:

$$F_{collision}(x_{crush}) = k_{body} * x_{crush} + F_o \tag{22}$$

Where $F_{collision}(x_{crush})$	is the collision force as a function of crush,
k_{body}	is the linear spring constant representing the stiffness of the vehicle body frame,
x_{crush}	is the amount of crush of the body frame,
F_o	is the breakout force.

The breakout force is an initial body frame rigidity that must be overcome to cause permanent deflection of the frame. Another nonlinearity comes from the fact that the crush is a residual crush and thus no spring recovery is allowed. In addition to the nonlinearities, the model is further complicated in that it has no dynamics if a collision does not generate forces greater than the breakout force. In the derivation of the Strother model, it is shown

that no residual deformation takes place if the energy of the collision is less than the five mile per hour kinetic energy level of the vehicle. It is from this energy level that the breakout force is derived. This jump in energy required for deformation occurs because energy is absorbed by the bumper. To be able to model collision dynamics for low energy level collisions, a separate bumper model is required.

3.1.3 Bumper Model

The bumper model is used when the change in kinetic energy of the vehicles during a collision (as calculated by conservation of momentum) is below the five mile per hour energy level of the lighter of the two vehicles involved in the collision. The bumpers are modeled as simple spring-dampers. The damping ratio and spring rate are selected so as to stop the mass of the vehicle at a deflection of 0.15 m when the mass is given an initial velocity of 2.24 m/s, and to provide an overdamped response.

3.1.4 Model Validation

After completion of the model it was validated in two stages. First, the dynamic properties of the model were verified, then simulated residual crushes were checked against actual crush data. To check the dynamical properties of the model, a two car collision was simulated. The final velocity predicted by the model was checked against final velocity predicted by a balance of momentum calculation, and the interaction forces were examined to see if they balanced at all times. The final velocities were found to coincide, as were the interaction forces. In checking residual crush, it was found that an additional parameter was needed to match the actual crash data. In collecting the crash data, initial vehicle length measurements were made to the end of the bumper. Thus the post collision measurements included both the vehicle residual crush, as well as the bumper crush. In the model, dynamic crush forces are calculated from the point of zero residual crush. To account for the bumper crush, the expression for the collision force is modified as follows:

$$F(x_{crush}) = k(x_{crush} - x_b) + F_o \tag{23}$$

where x_b is the active bumper length. As the database did not provide the active length of the bumper, this parameter was determined for each car. Setting the active bumper length between 0.10 m and 0.25 m brought the predicted residual crushes to within 2% of the actual values.

3.2 Detecting Contact Between Vehicles

To establish when contact between two vehicles occurs, the shape and the positions of the cars are tracked at all times. Once a collision occurs, the vehicles will deform. To account for these deformations, the cars are allowed to crush in a controlled manner at the front and rear. The bumper's contact surface remains linear, generating a crushed vehicle planform in the shape of a trapezoid. Therefore, the final shape of a crushed vehicle will be trapezoidal. Figure 4 shows a vehicle before and after a collision.

In the current program, a collision is defined by observing when the areas of any of the cars intersect. The area that the vehicle sprung mass occupies is defined as the area of a trapezoid (in the x-y plane) whose centerline has the same yaw angle as the vehicle. Full three dimensionality of the vehicles has not yet been accounted for in the collision detection algorithm.

4 Simulation Program

The current version of the program is capable of simulating a platoon consisting of an arbitrary number of cars traveling along any given road profile with no super-elevation (no banked roads). The code has been constructed within the SIMULINK [The MathWorks, Inc. 1994] framework. The individual modules for the engine, torque converter, steering dynamics,

brake dynamics, aerodynamics, collision dynamics, and tire model have been programmed in subroutines as MATLAB [The MathWorks, Inc. 1994] functions, S-functions, CMEX functions, and C functions. The detailed documentation of the code is contained in Appendix B.

Before running the simulation, various physical parameters (such as mass and size of the cars), road profile, and initial conditions can be specified from an interactive user interface. Also, for initial analysis, the complexity of the dynamics model is variable. For example, if the user wishes to see if a collision will occur given certain initial conditions, he or she can choose a dynamic model which only includes the planar dynamics of the vehicle and choose to disable the routines which calculate the collision forces. If a collision does occur, the user can opt for a more detailed analysis by including the routines which calculate the collision forces.

During nominal operations, numerical integration of the equations of motion and collision detection tests are performed simultaneously. Control input forces, aerodynamic drag forces, and road-tire interaction forces are taken into account in the equations of motion. Once a collision is detected, a separate subroutine determines the impact forces between the interacting vehicles during the collision. These impact forces are added into the vehicle dynamics routines.

5 Summary

A modular simulation program has been developed to study the platoon dynamics during both nominal and emergency situations on the road. The vehicle model is a three dimensional model in which the sprung mass has the full six degrees of freedom. Also, models for the engine, transmission, aerodynamics, and tire-road interactions have been included in modules which the user may also change if desired. The program is structured such that the user can customize the platoon models and implement different controllers. Therefore, this program

lends itself to be used as a tool for design change evaluations.

APPENDIX

A Gear Shifting Routine

The gear shifting routine was developed by Paul Sachi, Dynamics Laboratory, UC Berkeley. The inputs to the gear shifting subroutine are the car's position number in the platoon and the engine speed. The currently engaged gear is obtained from a matrix loaded from the MATLAB workspace. From the car's position number, the engine type is extracted from ENGNUM_mat, engine number data matrix obtained from NHTSA data. The engine speed is used to determine whether or not it exceeds the maximum speed for the currently engaged gear. If the engine speed exceeds this maximum speed, the gear shifting routine will shift up the gear. (The gear will not shift up if the car is in its highest gear.) If the engine speed is below the down-shifting speed (specified by maximum engine speed for gear "0"), the routine will downshift. (The gear will not downshift if the first gear is engaged.) The data for these maximum engine speeds came from the BMW M3 engine, the 1994 Chevy Cavalier engine, the 1994 Chevy Monte Carlo Z34 engine, and the 1994 chevy Impala SS engine. The engine properties for a Ford Lincoln Town car engine have been obtained from [McMahon1994].

The output of the gear shifting routine is the transmission reduction (r-star), which is defined as $1/d_{ratio}$. The d_{ratio} is a product of the engaged gear ratio and the gear reduction factor.

Table 2: Maximum Engine Speeds

Engine Type	Gear Number	Max Engine Speed for the Gear[RPM]
BMW M3	0	1500
"	1	6800
"	2	6800
"	3	6500
"	4	6500
Chevy Cavalier	0	1500
"	1	6000
"	2	6000
"	3	6000
"	4	5800
Chevy Monte Carlo	0	1500
"	1	7000
"	2	7000
"	3	5400
Chevy Impala	0	1000
"	1	5500
"	2	5500
"	3	5500
Lincoln Town Car	1	1000
"	1	5500
"	2	5500
"	3	5500

Table 3: d_{ratio}

Engine Type	Gear Number	d_{ratio}
BMW M3	1	13.230
"	2	7.844
"	3	5.229
"	4	3.906
"	5	3.150
Chevy Cavalier	1	13.998
"	2	7.679
"	3	5.191
"	4	3.687
"	5	2.649
Chevy Monte Carlo	1	10.020
"	2	5.385
"	3	3.430
"	4	2.400
Chevy Impala	1	9.425
"	2	5.020
"	3	3.080
"	4	2.156
Lincoln Town Car	1	7.848
"	2	4.797
"	3	3.270
"	4	2.181

B Simulation Code Documentation

The files described in this appendix are organized by functionality. The sections are as follows: Actuators, Aerodynamics, Car Body Dynamics, Controllers, Data, Engine, Initialization, Interpolation, Tire Dynamics, Transmission, and Main Program Support Files.

There are five types of files: MATLAB function, S-function, CMEX function, C function, and header files for C functions. A MATLAB function is a function written in MATLAB syntax. An S-function is a function which allows for internal dynamics in the SIMULINK framework. (The syntax is identical to the MATLAB syntax.) A CMEX function is a function written in C and also contains an interface to MATLAB. It is compiled with "cmex" to generate an executable file which can be called directly from MATLAB or SIMULINK. A C function is a function written in C which is usually called only from a CMEX or other C functions. A header file is a file containing information for the C preprocessor. If certain preprocessor information is required by several different C or CMEX functions, it is usually contained in the header files.

B.1 Variable List

The following is a list of the variables as they appear in the programs.

alead: acceleration in the x-direction of the lead vehicle

alpha: actuated throttle angle in degrees

alpha-dot: time derivative of alpha in degrees/s

BODY-DIM-mat: matrix containing constants (HO, H2, H4, H5, L1, L2, SB, DS) associated with the body of the vehicle (aside from the length and the width of the vehicle) for each vehicle in the platoon

carnum: the number of the car in the platoon; the numbering starts with the first car behind the lead car

car-type: the number corresponding to the row of the NHTSA database which contains the data on the vehicle specified by car-type; car-type is determined by ModelNo and MakeNo

CAR-TYPE-mat: matrix containing the car-type for each vehicle in the platoon

celica: the number car-type for the Toyota Celica

ENGNUM-mat: matrix containing the engine number of for each vehicle in the platoon the engine number usually refers to the number of cylinders in the engine

fp: the forces in the z-direction exerted on the four tires of a vehicle (N)

F_aero: aerodynamic force vector in Newtons (N)

F-collision: collision force vector in Newtons (N)

F-tire: tire-road interaction force vector in Newtons (N)

I-xyz-mat: matrix containing the moment of inertia about the x-, y-, and z-directions for each vehicle in the platoon

K-now: 1/rho, where rho is the radius of curvature for the road in meters

LENGTH_mat: matrix containing the vehicle length for each vehicle in the platoon

M_aero: aerodynamic moment vector in Newton-meters (Nm)

M-collision: collision moment vector in Newton-meters (Nm)

M-tire: tire-road interaction moment vector in Newton-meters (Nm)

ma: mass of air in manifold in kg

ma-dot: time derivative of ma in kg/s

MakeNo: number in the NHTSA database which corresponds to a given make of a vehicle; examples of makes are Honda, Chevrolet, Ford, etc.

MakeNo_celica: MakeNo corresponding to the Toyota Celica

MASS-mat: matrix containing the vehicle mass for each vehicle in the platoon

ModelNo: number in the NHTSA database which corresponds to a given model of a vehicle; an example of a model is Accord (for the make Honda)

ModelNo_celica: ModelNo corresponding to the Toyota Celica

mom_x: sum of the moments about the x-direction in Nm

mom-y: sum of the moments about the y-direction in Nm

mom_z: sum of the moments about the z-direction in Nm

plead: position in the x-direction of the lead vehicle

pos-x: position in the x-direction for the given vehicle in meters

pratio: pressure ratio (manifold/atmospheric pressure)

pri: pressure influence function

r-star: transmission reduction

steerF: actuated steering angle for the front tires in degrees

steer-c: steering angle commanded by the controller in degrees

steerF_dot: time derivative of steerF in degrees/s

sum-f-x: sum of the forces in the x-direction in N

sum-f-y: sum of the forces in the y-direction in N

sum_f_z: sum of the forces in the z-direction in N

SUSPENSION-mat: matrix containing constants (WBAR, Cl, C2, D1, D2) associated with the suspension for each vehicle in the platoon

t: time in seconds (s)

t-brake: actuated brake torque **in** Nm

t-brake-c: brake torque commanded by the controller in Nm

t-brake-dot: time derivative of t-brake in Nm/s

t-pump: pump torque of the torque converter in Nm

TAU_BRAKE_mat: matrix containing the brake actuator time constant for each vehicle in the platoon

TAU_STEER_mat: matrix containing the steering actuator time constant for each vehicle in the platoon

tc: throttle characteristic

TIRE-CHAR-mat: matrix containing constants (Frollfrict, J-FWHEEL, J_RWHEEL, Kf, Kt, R_ORIG) associated with the tire dynamics for each vehicle in the platoon

tire-slip-angle: slip angles of the four tires (radians)

tire-steer-angle: steering angle inputs to the four tires (radians)

tire-vel: linear velocities of the four tires (m/s)

THROTTLE-mat: matrix containing constants (TAU_THROTTLE, MAX-THROTTLE, MIN_THROTTLE, MAX-THROTTLE-RATE) associated with the throttle for each vehicle in the platoon

vlead: velocity in the x-direction of the lead vehicle

w_eng: engine speed in radians/s

w_eng_dot: time derivative of w_eng in rad/s^2

WIDTH-mat: matrix containing the vehicle width for each vehicle in the platoon

x: the seventeen states of the vehicle

xdot: time derivative of x (car states)

xprev: information (position, velocity, acceleration in the x-direction) about the car directly in front of the given vehicle

B.2 Actuators

Sbrake.m: S-function; calls brake-dot.m to get brake actuator dynamics.

input (2): t-brake-c, carnum

output (1): t-brake

brake-dot.m: MATLAB function; contains brake actuator dynamics.

input (4): t, t-brake, t-brake-c, carnum

output (1): t-brake-dot

Ssteer.m: S-function; calls steer-dot.m to get steering actuator dynamics.

input (1): steer-c, carnum

output (1): steerF

steer-dot.m: MATLAB function; contains steering actuator dynamics.

input (3): t, steerF, steer-c, carnum

output (1): steerF_dot

Sthrottle.m: S-function; Calls throttle-dot.m to get throttle actuator dynamics.

input (1): alpha-c, carnum

output(1): alpha

throttle-dot.m: MATLAB function; contains throttle actuator dynamics.

input (3): t, alpha, alpha-c, carnum

output (1): alpha-dot

B.3 Aerodynamics

aero-M.m: MATLAB function; contains aerodynamic model developed using data on platoon aerodynamics from [Zabat et al. 1995].

input (3): x, xprev, carnum

output (2): F-aero, M_aero

B.4 Car Body Dynamics

Scarbody.m: S-function; calls carbody_dot.m to get car dynamics.

input (7): sum-f-x, sum-f-y, sum-f-z, mom_x, mom-y, mom_z, carnum

output (1): x

carbody.h: Contains "#define" statements which define what all the states in the car state vector, x, represent.

carbody_dot.c: CMEX file; contains dynamic model for the car body.

input (9): t, x, sum_f_x, sum_f_y, sum_f_z, momx, mom-y, mom_z, carnum

output (1): xdot

sum-fandm.m: Used to sum total forces in each of the coordinate directions and to sum total moments about each of the coordinate directions. This is used as a MATLAB Function Block in SIMULINK because it is simpler than drawing out all the summing blocks, input blocks, output blocks, etc.

input (6): F_aero, M_aero, F-tire, M-tire, F-collision, M-collision

output (6): sum_f_x, sum-f-y, sum_f_z, momx, mom-y, mom_z

B.5 Controllers

control-var.11: Header file; contains type-structure definitions for control variables.

Slat_controller.m: S-function; calls lat_controller.c to get lateral controller dynamics.

lat-control1er.c: CMEX function; contains the lateral controller model.

Slong-control1er.m: S-function; calls long-control1er.m to get longitudinal controller dynamics.

long-control1er.m: MATLAB function; contains the longitudinal controller model.

slide.c: C function; called by lat_controller.c.

tcinv.m: MATLAB function; called by long-control1er.m. (It is the inverse of the tc.m function.)

input (1): tc

output (1): alpha

B.6 Data

The directory called "data" contains two subdirectories which contain data tables pertinent to the simulation. These subdirectories are "car-models-data" and "ford-engine" directories.

B.6.1 car-models-data directory

makemodelstrings.mat: contains two matrices of strings which list the makes and the models of the vehicles in the database.

models-dat.mat: Data from NHTSA which contains data pertaining to several vehicle models. The data used from this source are the length, mass, width, and the number of cylinders in the engine

B.6.2 ford-engine directory

ford2_dat.mat: contains the Ford Lincoln Town car engine raw data which is processed by read-engdat.m

read-engdat.m: Organizes the engine map data in "ford2_dat.mat" into the matrices we-table, p-table, tnet-table, mao_table, and p-table-entry-count.

tc_dat.mat: Contains throttle characteristics data. The first column contains the throttle angle, and the second column contains the corresponding throttle characteristics.

tq1_dat.mat: Contains torque converter data for when the engine is driving the wheels. The first column is "spr", second column is "tpr", and the third column is "cpr". "spr" is the wheel to engine speed ratio, where the engine speed is adjusted by the transmission reduction, r_star. "tpr" is the turbine to pump torque ratio. "cpr" is the capacity ratio, which is defined as the engine speed divided by the square root of the pump torque. The first column is spr, second column is tpr, and the third column is cpr.

tq2_dat.mat: Contains torque converter data for when the wheels are driving the engine. The first column is spr, second column is tpr, and the third column is cpr.

B.7 Engine

Sengine.m: S-function; calls engine-dot.m to get engine dynamics.

input (3): alpha, t-pump, carnum

output (1): w-eng

engine-dot .m: MATLAB function; contains two-state model for a Ford engine. Calls tc.m, pri.m, interpo12.m.

input (6): t, ma, w-eng, alpha, t-pump, carnum

output (2): ma-dot, w-eng-dot

engine_h.m: MATLAB function; contains some physical constants and engine parameters which used to be contained in engine.11 of the VDL code.

pri.m: MATLAB function; called from engine-dot.m. input (1): pratio

output **(1):** pri

tc.m: MATLAB function; called from engine-dot.m. (It is the inverse of the tcinv.m function.) input (1): alpha

output(1): tc

B .8 Initialization

These are files used to set up the initial conditions for the simulation.

choose-mode1s.m: MATLAB function; allows the user to specify (from the NHTSA database) the vehicle models to be used in the platoon. This function outputs five matrices which contain the information obtained from the NHTSA database. It is called from init.m.

input (3): MakeNo_celica, ModelNo_celica, celica

output (5): MASS mat, WIDTH-mat, LENGTH-mat, ENGNUM_mat, CAR-TYPE-mat

desc2.m: MATLAB function; given the make of a vehicle, this function outputs the model numbers and the associated model names in the database. This function also has an error check on the MakeNo that the user specifies. It is called from choose-mode1s.m

input (1): MakeNo

output (3): model_nos, modelaames, Make-new

desc3.m: MATLAB function; given the MakeNo and the ModelNo, this function outputs a number which corresponds to a set of data in the NHTSA database. It is called from choose-mode1s.m and init.m.

input (2): MakeNo, ModelNo

output (1): car-type

get-celica-data-m: MATLAB function; obtains the data for a Toyota Celica, which serves as the default choice for the vehicle models. It is called from init.m.

input (1): celica

output (7): TAU_BRAKE_mat, TAUSTEER-mat, THROTTLE-mat, SUSPENSION_mat, TIRE-CHAR-mat, BODY_DIM_mat, I_xyz_mat

get-init-p0s.m: MATLAB function; sets up the initial positions of the platoon vehicles in the desired coordinate system.

get_init_vel.m: MATLAB function; sets up the initial velocities of the platoon vehicles in the desired coordinate system.

init.m: MATLAB function; it is called from main-prog.m. Sets up initial conditions for the platoons, such as the vehicle positions and velocities. This routine calls several functions to accomplish the task of initialization.

print-MAKE.m: Prints the makes of the vehicles that are available in the database. This is used when the user is customizing the vehicle types in the platoon. It is called from choose-mode1s.m.

B.9 Interpolation

interpol.c: C function; called from This function is a one-dimensional interpolation routine.

interpo12.m: MATLAB function; called from This function is a two-dimensional interpolation routine.

interpol_mx.c: CMEX function; same function as interpol.c, for use in interpo12.m and other MATLAB functions requiring one-dimensional interpolation.

locate_mx.c: CMEX function; called from interpo12.m.

B.10 Tire Dynamics

suspension-mx.c: CMEX file; contains suspension model.

input (1): x

output (1): fp

slip-ang1e.c: C-code file; computes **the** slip angle for tire force calculations.

input: x, tiresteer-angle

output: tireslip-angle, tire-vel

tire-forces-mx.c: CMEX file; compute total forces and moments on the car due to the forces at the tire/road interface. This function calls slip-angle.c, tire_model.c (which contains the Bakker-Pacejka tire model).

input (8): tiresteer-angle, tireslipangle, w-wheel, tire-vel, pos_z, phi, theta, fp

output (7): mx,my,mz, sum_fa, sumfb, sum_fp, t-tractive

Swheel.m: S-function; calls wheel-dot.m to get the wheel dynamics.

input (5): carnum, t-brake, r-star, t-turb, t-tractive

output (4): w-wheel

wheel-dot.c: CMEX file; contains dynamic model for the wheels.

input (6): t, w-wheel, t-turb, r-gear, t-brake, t-tractive

output (1): derivatives of w-wheel

B.l 1 Transmission

B.ll.l Gear Shifting

gear-shift-mx.c: CMEX file; contains gear shifting routine.

input (2): x, w-eng

output (1): r-star

(also, the matrix containing the engaged gears is MODIFIED in this program.)

B.11.2 Torque Converter

torq_conv_mx.mexds: CMEX file; contains the torque converter model.

input (4): r-star, w-wheels, w-eng, carnum

output (2): t-turb, t-pump

B.12 Main Program and Main Program Support Files

lead-profi1e.m: MATLAB function; Generates the lead car velocity and acceleration profiles.

input (1): t

output (3): alead, vlead, xlead

main-pr0gram.m: MATLAB script file; the main program. Declares and defines global variables. Calls init.m to choose vehicle models and set up initial conditions. Sets up engine and torque converter data tables. Runs the simulation.

roadcurve.c: Returns the road curvature for every section of the simulated road.

input (1): pos-x

output ():K_now

References

[Dugoff 1969] Howard Dugoff, Paul S. Fancher, and Leonard Segel. Tire Performance *Characteristics* Affecting Vehicle Response to Steering and Braking Control Inputs Michigan Highway Safety Research Institute, Ann Arbor, Michigan, 1969.

[Hedrick et al. 1991] J.K. Hedrick, et al. "Longitudinal vehicle controller design for IVHS systems." Proceedings of *the* 1991 American Control *Conferen ce.*

[McMahon 1994] Donn H. McMahon. Robust Nonlinear Control of Uncertain Systems: An Application to Intelligent Vehicle *Highway* Systems *(IVHS)*, Ph.D. dissertation, UC Berkeley, 1994.

[The MathWorks, Inc. 1994] The MathWorks, Inc. SIMULINK, MATLAB, Version 4.2a, 1994.

[Peng 1992] Huei Peng. Vehicle Lateral Control for *Highway* Automation, Ph.D. dissertation, UC Berkeley, 1992.

[Peng et al. 1991] H. Peng and M. Tomizuka. "Preview control for vehicle lateral guidance in highway automation." Proceedings of the 1991 American Control Conference, 2621-2627.

[Pham 1995] Masayoshi Tomizuka, J. Karl Hedrick, and Hung Pham. "Integrated Maneuvering Control for Automated Highway Systems Based on a Magnetic Reference/Sensing System." PATH Re*search* Report UCB-ITS-PRR-95-12, 1995.

[Sheikholeslam 1989] S. Sheikholeslam and C.A. Desoer. "Longitudinal control of a platoon of vehicles, I: linear model." PATH Research Report UCB-ITS-PRR-89-3, 1989.

[Strother et al. 1987] Charles E. Strother, Ronald L. Woolley, Michael B. James, and Charles Y. Warner. Collision Safety Engineering, Inc. "Crush Energy in Accident Reconstruction." ***SAE Paper 860371,*** 1987.

bibitem[Tongue 1995]tongue Benson H. Tongue and Ahrie Moon. "Low Speed Collision Dynamics and Control: Year One Report." PATH ***Research Report*** UCB-ITS-PRR-95-25, 1995.

[Wong 1993] J.Y. Wong. ***Theory of*** Ground Vehicles, John Wiley and Sons, 1993.

[Zabat et al. 1995] Michael Zabat, Nick Stabile, Stefano Frascaroli, and Fred Browand. "Drag Forces Experienced by 2,3, and 4-Vehicle Platoons at Close Spacings." ***SAE*** paper No. 950632, 1995, pp. 122-123.