Journal of Mechanical Science and Technology, Vol. 19, No. 1, 〈Special Edition〉 pp. 395~402, 2005 395

A Symbolic Computation Method for Automatic Generation of a Full Vehicle Model Simulation Code for a Driving Simulator

Ji-Young Lee*
S&TRI, InnoSimulation, Inc.,
Donggyo-dong, Mapo-gu, Seoul 121-819, Korea
Woon-Sung Lee
Graduate School of Automotive Engineering, Kookmin University,
Seoul 136-702, Korea

This paper deals with modeling and computer simulation of a full multibody vehicle model for a driving simulator. The multibody vehicle model is based on the recursive formulation and a corresponding simulation code is generated automatically from AUTOCODE, which is a symbolic computation package developed by the authors using MAPLE. The paper describes a procedure for automatically generating a highly efficient simulation code for the full vehicle model, while incorporating realistically modeled components. The following issues have been accounted for in the procedure, including software design for representing a mechanical system in symbolic form as a set of computer data objects, a multibody formulation for systems with various types of connections between bodies, automatic manipulation of symbolic expressions in the multibody formulation, interface design for allowing users to describe unconventional force- and torque-producing components, and a method for accommodating external computer subroutines that may have already been developed. The effectiveness and efficiency of the proposed method have been demonstrated by the simulation code developed and implemented for driving simulation.

Key Words : Symbolic Computation, Mutibody Vehicle Model, Driving Simulator, Automatic Code Generation, Vehicle Dynamics, Real-time Simulation

1. Introduction

A driving simulator is a virtual reality tool that gives a driver on board impression that he/she drives an actual vehicle by predicting vehicle motion caused by driver input and feeding back corresponding visual, motion, audio and proprioceptive cues to the driver (Lee, 1998). The driving simulator thus requires real-time calculation of vehicle dynamics in response to driver input, such as steering, and accelerator and brake pedal operation.

Two fundamental requirements of vehicle dynamic simulation for the driving simulator are that it should execute faster than real time and that it should provide high fidelity. These two requirements are conflicting in that higher simulation fidelity requires greater model complexity and longer run times. The nature of trade-off between high fidelity and computational efficiency has not changed significantly in spite of rapid development of computing hardware, especially PCs.

Lumped parameter vehicle models with reduced degrees-of-freedom have been adopted by most of driving simulators around the world because of the conflicting requirements mentioned

---

* Corresponding Author,
E-mail : jylee@innosim.com
TEL : +82-2-337-5001; **FAX** : +82-2-337-6440
S&TRI, InnoSimulation, Inc., Donggyo-dong, Mapo-gu, Seoul 121-819, Korea. (Manuscript **Received** November 29, 2004; **Revised** December 15, 2004)

above (Weir, 1995). On the other hand, multibody dynamic models have been adopted by few simulators including NADS (Salaani, 1997), VIRTTEX (Artz, 2002) and DaimlerChrysler Driving Simulator (Kading, 1995).

There are some advantages of using multibody vehicle models for the driving simulators. The multibody vehicle models can first take the effects of suspension geometry into consideration. With the multibody dynamic models it becomes easy to modify the suspension geometry and to estimate its effects without preparing a complex suspension compliance characteristics map. Another advantage seems indispensable for realizing a virtual proving ground by the driving simulator. This advantage is as follows: Parameters of an actual vehicle can correspond to the parameters of a multibody vehicle model in one-to-one. However, a simplified vehicle model requires parameter conversion from actual vehicle parameters to inherent model parameters, for example, roll stiffness. The multibody dynamic model does not require parameter conversion, so that it is easy to change the parameter setting of the vehicle. To enhance the performance of the vehicle, many combinations of parameters should be examined by the proving ground, whether it is actual or virtual. Thus this advantage seems to have great significance.

To solve this problem, the authors propose a symbolic computational method that can generate a full vehicle multibody model simulation code automatically for a driving simulator. A symbolic method is presented for expressing the equations of motion in explicit form, eliminating the need for numerically solving sets of simultaneous equations. For the vehicle system, the symbolic method can be much more efficient than numerical methods commonly used. The efficiency of the symbolic computational method has ever been verified (Choi, 2000). The AUTOCODE, a symbolic computation package using MAPLE (Watcom, 1991), has been developed to generate the multibody dynamic equations in symbolic form automatically and convert them into a C or Fortran code for implementation on the real-time computer. It is shown in the following sections that the proposed method using AUTOCODE realizes real-time dynamic simulation with the multibody full vehicle model and has sufficient accuracy of calculation. The efficiency and effectiveness of the vehicle dynamic model is demonstrated through its implementation on a newly developed driving simulator system, RTSA DS.

2. Symbolic Processing of Multibody Vehicle Dynamics

The main objective of this work is to create a means for automatically generating highly efficient simulation codes for ground vehicles, while incorporating realistically modeled components. For this, the paper includes (1) software design for representing a mechanical system in symbolic form as a set of computer data objects, (2) a multibody formalism valid for systems with various types of connections between bodies, (3) methods to manipulate symbolic expressions automatically within the multibody formalism, (4) design of an interface for simple description of unconventional force- and torque-producing components, and (5) a method to accommodate external computer subroutines that may have already been developed separetely.

2.1 Multibody formulation

In deriving dynamic equations for general multibody models, the variational approach has been adopted in this study. Relative generalized coordinates have been chosen to represent motion of bodies in the system (Lee, 1994).

The variational Newton-Euler equations of motion in relative generalized coordinates are given as

$$\delta q^T(\bar{M}\ddot{q}-\bar{Q})=0 \qquad (1)$$

where q represents relative generalized coordinates defined in a vehicle model, $\bar{M}$ indicates generalized mass matrix, and $\bar{Q}$ generalized forces directly applied and due to control action on a body. The relative variation δq must be consistent with constraint equations obtained by cutting a closed kinematic loop.

$$\Phi(q)=0 \qquad (2)$$

This implies that the relative coordinates q are not all independent. Dividing q into dependent and independent coordinates, that is, $q=[u, v]^T$, Eq. (1) and Eq. (2) can be rewritten as

$$[\delta u^T, \delta v^T]\left\{\begin{bmatrix}\bar{M}_{uu} & \bar{M}_{uv}\\ \bar{M}_{vu} & \bar{M}_{vv}\end{bmatrix}\begin{bmatrix}\ddot{u}\\ \ddot{v}\end{bmatrix}-\begin{bmatrix}\bar{Q}_u\\ \bar{Q}_v\end{bmatrix}\right\}=0 \quad (3)$$

$$\Phi(u, v)=0 \quad (4)$$

δu and δv in Eq. (3) must be consistent with the constraint Eq. (4). The Lagrange multiplier theorem (Haug, 1989) guarantees existence of a Lagrange multiplier vector λ such that

$$[\delta u^T, \delta v^T]\left\{\begin{bmatrix}\bar{M}_{uu} & \bar{M}_{uv}\\ \bar{M}_{vu} & \bar{M}_{vv}\end{bmatrix}\begin{bmatrix}\ddot{u}\\ \ddot{v}\end{bmatrix}+\begin{bmatrix}\Phi_u^T\\ \Phi_v^T\end{bmatrix}\lambda-\begin{bmatrix}\bar{Q}_u\\ \bar{Q}_v\end{bmatrix}\right\}=0 \quad (5)$$

Taking the first order variation of Eq. (4) and solving for δu, we obtain

$$\delta u=-\Phi_u^{-1}\Phi_v\delta v\equiv R_1\delta v \quad (6)$$

Similarly, taking the time derivative of Eq. (4) twice and solving for $\ddot{u}$, we obtain

$$\ddot{u}=-\Phi_u^{-1}\Phi_v\ddot{v}-\Phi_u^{-1}(\dot{\Phi}_u\dot{u}+\dot{\Phi}_v\dot{v})\equiv R_1\ddot{v}+R_2 \quad (7)$$

Substituting Eq. (6) and Eq. (7) into Eq. (5) and setting the coefficient term of δv equal to zero, the dynamic equations of the system can be obtained in terms of the independent coordinates as

$$\hat{M}\ddot{v}=\hat{Q} \quad (8)$$

where

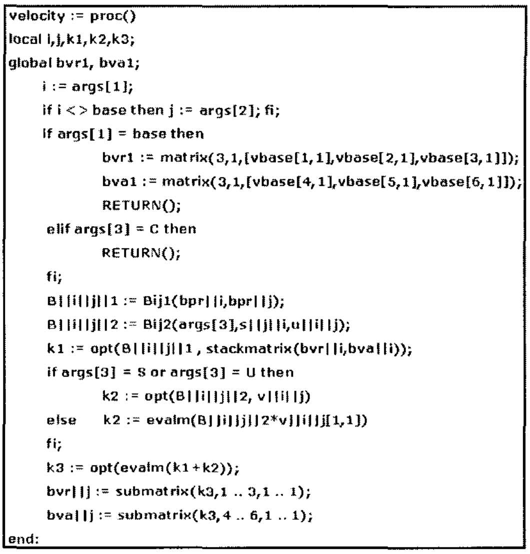
$$\hat{M}=R_1^T\bar{M}_{uu}R_1+R_1^T\bar{M}_{uv}+\bar{M}_{vu}R_1+\bar{M}_{vv} \quad (9)$$

$$\hat{Q}=R_1^T\bar{Q}_u+\bar{Q}_v-R_1^T\bar{M}_{vv}R_2-\bar{M}_{vu}R_2 \quad (10)$$

Therefore, a vehicle system is represented by Eqs. (8) ~ (10).

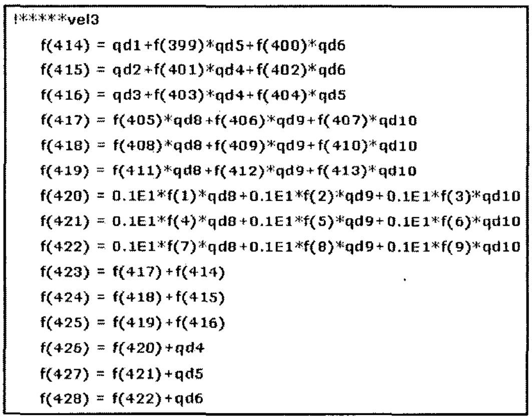
2.2 Symbolic computation

The MAPLE software has been used to develop procedures for symbolically computing velocity transformation matrices and terms for virtual displacement, velocity, and acceleration relationships between dependent and independent coordinates. For example, Fig. 1 shows a body velocity computation procedure in AUTOCODE. Fig. 2 shows a symbolic expression of the fifth body velocity obtained by executing the procedure in Fig. 1 with parameters and variables defined. In a similar fashion, the dynamic equations in Eq. (8) have been obtained in a symbolic form. This form has been then transformed into Fortran.



```
velocity := proc()
local i,j,k1,k2,k3;
global bvr1, bva1;
     i := args[1];
     if i < > base then j := args[2]; fi;
     If args[1] = base then
               bvr1 := matrix(3,1,[vbase[1,1],vbase[2,1],vbase[3,1]]);
               bva1 := matrix(3,1,[vbase[4,1],vbase[5,1],vbase[6,1]]);
               RETURN();
     elif args[3] = C then
               RETURN();
     fi;
     B||i||j||1 := Bij1(bpr||i,bpr||j);
     B||i||j||2 := Bij2(args[3],s||j||i,u||i||j);
     k1 := opt(B||i||j||1 , stackmatrix(bvr||i,bva||i));
     if args[3] = S or args[3] = U then
               k2 := opt(B||i||j||2, v||i||j)
     else      k2 := evalm(B||i||j||2*v||i||j[1,1])
     fi;
     k3 := opt(evalm(k1+k2));
     bvr||j := submatrix(k3,1 .. 3,1 .. 1);
     bva||j := submatrix(k3,4 .. 6,1 .. 1);
end:
```

Fig. 1 Body velocity calculation procedure



```
!*****vel3
    f(414) = qd1+f(399)*qd5+f(400)*qd6
    f(415) = qd2+f(401)*qd4+f(402)*qd6
    f(416) = qd3+f(403)*qd4+f(404)*qd5
    f(417) = f(405)*qd8+f(406)*qd9+f(407)*qd10
    f(418) = f(408)*qd8+f(409)*qd9+f(410)*qd10
    f(419) = f(411)*qd8+f(412)*qd9+f(413)*qd10
    f(420) = 0.1E1*f(1)*qd8+0.1E1*f(2)*qd9+0.1E1*f(3)*qd10
    f(421) = 0.1E1*f(4)*qd8+0.1E1*f(5)*qd9+0.1E1*f(6)*qd10
    f(422) = 0.1E1*f(7)*qd8+0.1E1*f(8)*qd9+0.1E1*f(9)*qd10
    f(423) = f(417)+f(414)
    f(424) = f(418)+f(415)
    f(425) = f(419)+f(416)
    f(426) = f(420)+qd4
    f(427) = f(421)+qd5
    f(428) = f(422)+qd6
```

Fig. 2 Fifth body velocity symbolic code

One thing to note here is that, in the symbolic approach, a long linear code is generated. This has caused a problem on a CPU in terms of overflow or overload, and it had to be fixed in an ad hoc fashion. Thus, the proposed method created a function that can substitute every recomputable symbolic part with a new intermediate variable. Since constant parameters are computed at symbolic calculation stage, they are not substituted with the intermediate variables. With Fortran code output option ‘optimized’ in MAPLE, this function guarantees the optimized code.

For example, a linear symbolic code, such as (q1*q2+q3)*(q4*q5-q6)+(q1*q2+q3)*q7, can be replaced by f(1)*f(2)+f(1)*q7, and intermediate variables f(1) and f(2) can be used in other calculation. Trigonometric functions can also be simplified by introducing additional variables. When matrix multiplication occurs, some part of matrix entries can be constant and the other part can be long symbolic code. The proposed method searches all matrix entries at every step of calculation for the target code that will be changed to intermediate variables automatically.

2.3 AUTOCODE

A symbolic computation package, AUTOCODE, has been developed to generate automatically the simulation code based on the previous symbolic computation approach. The combined effect of these new techniques is significant in at least two ways : (1) the simulation codes generated are efficient for vehicle dynamic simulation, and (2) the input description prepared by a user is minimal and does not require knowledge of the formalism details.

Fig. 3 describes how the model conceived by the user is described in simple terms and translated into computer representation. Once the system is described, the equations of motion are developed automatically and a self- contained C or Fortran simulation code is generated.

Techniques are being developed for handling kinematic constraints in a more automated manner than before with symbolic multibody analysis. Fig. 4 shows a graphic user interface for AUTOCODE. As the user inputs only the model topology, properties of bodies, geometries of joints and few more parameters, a dynamic simulation code is automatically generated.

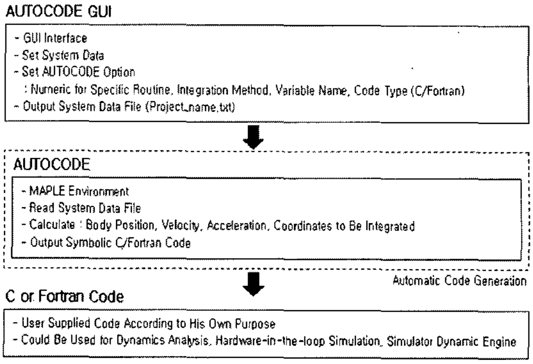
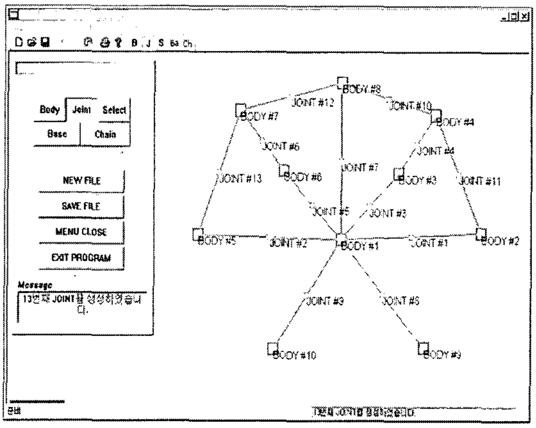


Fig. 3 Code generation procedure



13번째 JOINT를 생성하였습니다.

Fig. 4 AUTOCODE GUI

The codes generated by AUTOCODE are not completely general when it comes to introducing force- and torque-producing components. For example, ground vehicles include tires, nonlinear springs, complex shock absorbers, etc. that are modeled differently based on the intended use of simulation. Assuming that a user is able to develop computer representation of such an element as an external subroutine, the subroutine must be incorporated into the multibody simulation code. Variables needed as inputs to the external subroutine (positions, angles, speeds, etc.) are always readily available, by assigning a global variable type to these variables. Also force and torque computed by the external subroutine acts on the bodies as external force and torque.

3. Application to a Driving Simulator

The multibody vehicle model generated by AUTOCODE has been applied to a newly developed driving simulator as shown in Fig. 5. This is a driver-in-the-loop, high-fidelity real-time driving simulator for road traffic safety and human factor research (2004, Lee).

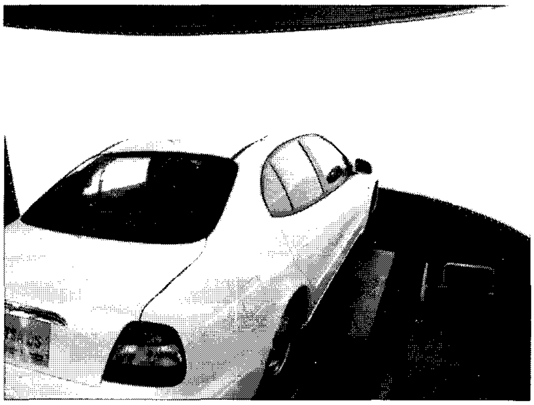


Fig. 5 Road traffic safety authority driving simulator

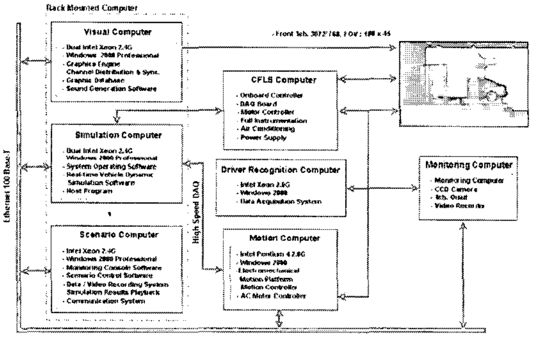


Fig. 6 Simulator Configuration

3.1 Road traffic safety authority driving simulator

The simulator consists of a real-time simulation system, a visual and audio system, a motion system, a control force loading system, and a console system. Fig. 6 shows a configuration of the simulator.

The real-time simulation system supervises overall operation of the simulator and also simulates dynamic motion of the developed multibody vehicle model in real time using SCANeR II. SCANeR II is comprehensive driving simulation software that provides functionality to control driving simulation scenarios including surrounding cars, surrounding traffic, weather and so on. Fig. 7 shows a scenario control example. The visual system generates high fidelity visual scenes that are displayed on a cylindrical screen and a flat rear screen by four LCD projectors. The motion system generates realistic motion cues using a six degree-of-freedom Stewart platform driven electrically. The control force loading system acts as an interface between a driver and the simulator, and provides the driver with steering wheel reaction torque and pedal reaction force. The console system monitors a status of the simulator in operation and driver's head and eye movement using FaceLAB (Seeing-Machines, 2003), and also collects and manages experimental data.

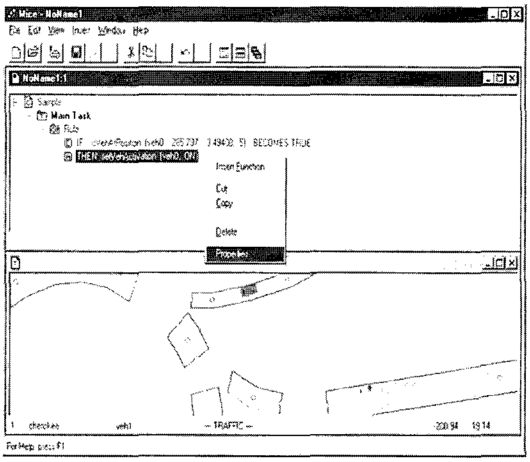


Fig. 7 Scenario control example

3.2 Vehicle dynamic modeling

Since a mid-size passenger cabin has been used for the RTSA driving simulator as shown in Fig. 5, a corresponding vehicle model has been developed accordingly. For the complete vehicle model, other submodels such as powertrain, brake, steering, tire, and drag have been added to the multibody vehicle model.

3.2.1 Multibody dynamic modeling

The multibody vehicle model is used to model the rigid body behavior of chassis, suspension, and steering rack. The vehicle model has front Macpherson strut suspensions and rear trailing arms, and is based on a topological analysis of a rigid body mechanism as shown in Fig. 8. The model contains four closed kinematic loops. A joint in each loop is cut to obtain a spanning tree. Table 1 summarizes the rigid bodies and the joints modeled.

Table 1 Rigid bodies and joints in vehicle model

Rigid Body Number	Body description
1	chassis
2	right front control arm
3	right front strut
4	right front spindle
5	left front control arm
6	left front strut
7	left front spindle
8	steering rack
9	right trailing arm
10	left trailing arm
Symbol	Joints
S	Spherical
R	Revolute
D	Distance constraint
T	Translational
----	Cut

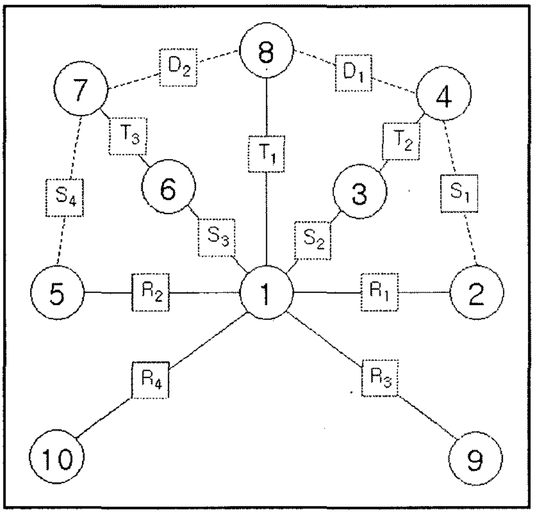


Fig. 8 Multibody model topology

The constraint library in AUTOCODE supports revolute, translational, universal, and spherical joints. For closed chains, distance constraint and spherical cut joints are supported. Also springs and dampers are modeled as external force elements. Nonlinear elements can also be used. The AUTOCODE will result in a model with optimum computational efficiency characterized by a minimum number of relative coordinates and constraint equations.

3.2.2 Vehicle subsystem modeling

The vehicle model also includes a complete set of subsystems such as powertrain, brake, steering, and tires. The subsystem models are added to the code generated by AUTOCODE and computations are completed for each of them at each simulation time step.

A powertrain subsystem with a spark-ignition engine and a four-speed automatic transmission is modeled. The powertrain model is based on one dimensional torque formulation that includes torque and angular velocity transmitted through each model component. These model components include an engine, a torque converter, an automatic transmission gearbox, a differential and a final drive.

A brake subsystem with front disc and rear drum brakes that are standard equipment with a fixed rear pressure-proportioning valve is modeled. Brake system dynamics is modeled by including a first-order lag between master cylinder pressure and pressure at the brake lines.

A steering subsystem with power-assisted steering is modeled. A nonlinear steering system model is developed for a rack-and-pinion steering system. Included in the model are friction, stiffness, damping characteristics of the steering system and a nonlinear boost curve.

A STI tire model (Allen, 1988) is used in the study. This model includes pertinent quasi-static tire dynamics as well as lateral and longitudinal lag dynamics.

3.3 Real-time simulation

It is difficult to realize real-time simulation with a high-order multibody vehicle model. One of the reasons of difficulty is that the equations of motion of the model are represented as differential-algebraic equations, so that the calculation of the constraint Jacobian and its inverse matrix are required at each time step of numerical integration.

In our vehicle dynamic analysis, the basic multibody model code by AUTOCODE is processed symbolically, meanwhile the final mass matrix inversion and subsystem model code are processed numerically. This approach can take advantages of better efficiency for both of symbolic computation of general terms and numeric computation of big matrix inversion.

Computation time has been measured using a PC with a PentiumIV 2.6 GHz CPU and a 1 G byte memory. It takes 0.51 ms to calculate one loop of the total vehicle dynamics code. Our proposed multibody analysis method realized real-time integration with 2 ms step size, and satisfactory stability of multibody dynamic calculation was obtained when the 3rd-order Adams-Bashforth method was used as a solver.

3.4 Simulation results

The developed dynamic model is checked against maneuvers that replicate real-world driving scenarios such as turning, lane-change, acceleration, and braking using the driving simulator. These simulation results represent driver-vehicle closed-loop performance. These simulation results have been saved by a monitoring computer in the simulator.

3.4.1 Step steer input

This maneuver is a step input of steering wheel angle at 12 m/s as shown in Fig. 9. It is used to determine steady state as well as transient response of a vehicle. The timing and peak levels of the vehicle response are reasonable.

3.4.2 Lane change maneuver

This maneuver is a lane change at 22.5 m/s as shown in Fig. 10. This maneuver is performed under a severe condition. Fig. 10 shows that the developed dynamic model simulates reckless driving faithfully.

3.4.3 Straight-line braking

This maneuver is straight-line braking at 22.2 m/s. Fig. 11 shows results from the severe braking run. The results show that the developed dynamic model predicted reasonable longitudinal deceleration. The brake line pressures were also predicted well by the brake model. A logic was added to wheel spin dynamics to include damping at the end of severe braking. The logic provides a smooth transition from a state of large longitudinal tire force to a state of relatively small force required once the vehicle comes to a complete stop.

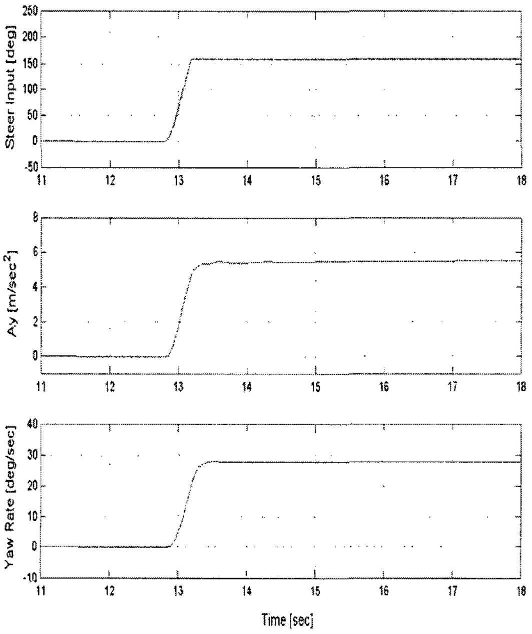


Fig. 9 Simulated step steer — 12 m/s

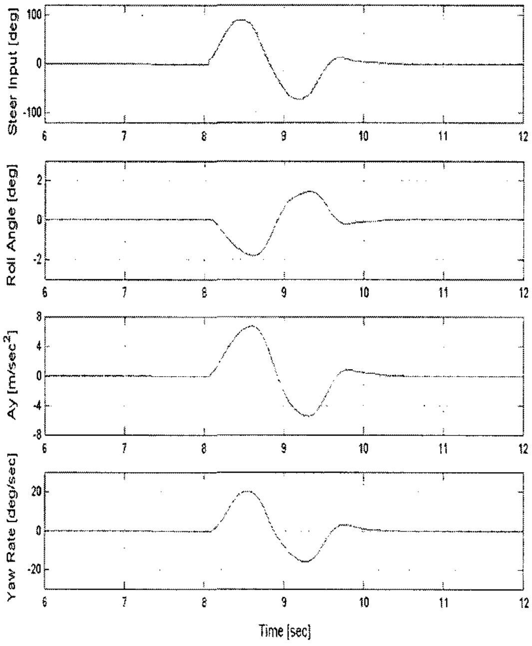


Fig. 10 Simulated lane change — 22.5 m/s

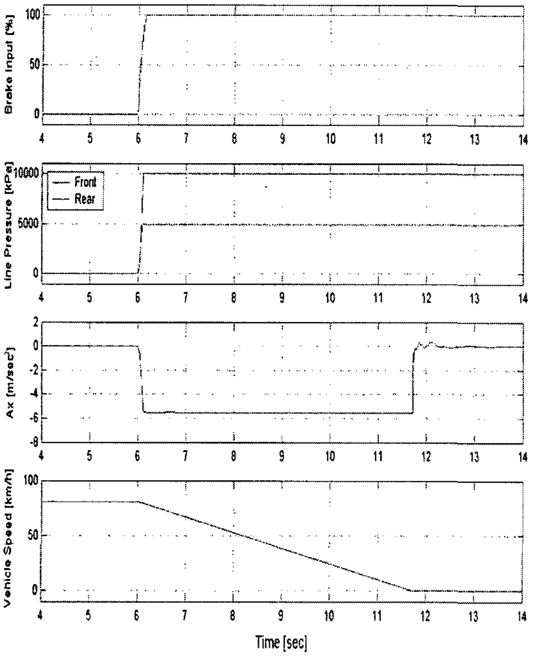


Fig. 11 Simulated severe braking

4. Conclusions

The fundamental requirements of vehicle dynamic simulation for the driving simulator are conflicting in that higher simulation fidelity requires greater model complexity and longer run times. To solve this problem, this paper proposed a symbolic computation method that can generate automatically a full vehicle multibody simulation code for a driving simulator. A symbolic computation package, AUTOCODE, has been developed based on the proposed method. The method has been applied to a newly developed driving simulator, by generating and executing a mutibody vehicle simulation code, and shown reasonable driver-vehicle performance under realistic driving conditions. Future works include expansion of AUTOCODE for more general applications.

References

Allen, R. W. and Rosenthal, T. J., 1988, *An Interactive Tire Model for Driver/Vehicle Simulation*, Final Report, DOT HS 807 271.

Artz, B. A., 2002, Personal Communication.

Choi, D. H. and Yoo, W. S., 2000, "Efficiency of a Symbolic Computation Method for the real Time Simulation," *Journal of KSME*, Vol. 24, No. 7, pp. 1878~1884. (in Korean)

FaceLAB User's Manual, SeeingMachines, 2002.

Haug, E. J., 1989, Computer Aided Kinematics and Dynamics of Mechanical Systems, Allyn and Bacon.

Kading, W. and Hoffmeyer, F., 1995, "The Advanced Daimler-Benz Driving Simulator," *SAE Technical Paper*, No. 950175.

Lee, W. S., Jung-Ha Kim and Jun-Hee Cho, 1998, "A Driving Simulator as a Virtual Reality Tool," *Pro. of the 1998 IEEE International Conference on Robotics & Automation*, Vol. 1, pp. 71~76.

Lee, W. S., 2004, The Development of a Driving Simulator, Final Report, Road Traffic Safety Authority. (in Korean)

Lee, W. S., Lee, J. S. and Kim, S. S., 1994, "Generation of Symbolic Equations of Motion for Real-Time Vehicle Simulation," *Proc. of KSME Fall Meeting*, pp. 327~331.

MAPLE V Language Reference Manual, Watcom, 1991.

Salaani, M. K., Heydinger, G. J. and Guenther, D. A., 1997, "Validation Results from Using NADSdyna Vehicle Dynamics Simulation," *SAE Technical Paper*, No. 970565.

Weir, D. H. and Clark, A. J., 1995, "A Survey of Mid-Level Driving Simulators," *SAE Technical Paper*, No. 950172.