

실시간 시뮬레이션을 위한 기호연산기법의 유용성 검증에 관한 연구

최 대 환* · 유 완 석**
(1999년 10월 30일 접수)

Efficiency of a Symbolic Computation Method for the Real Time Simulation

Dae-Hwan Choi and Wan-Suk Yoo

Key Words: Symbolic Computation(기호연산), Velocity Transformation Method(속도변환법), AutoDyn7(Automobile Dynamics in G7)

Abstract

In multibody dynamic analysis, one of the most important problems is to reduce computation times for real time simulation. In this paper, a symbolic computation method is implemented and tested for each dynamic analysis step. Applying symbolic formulations to the vehicle dynamics program AutoDyn7, the effectiveness of the symbolic computation method is verified.

1. 서 론

현재 다물체 동역학 시스템을 해석하고 설계하는데 CAE기법을 많이 이용하고 있으며, 이를 이용하여 다양한 컴퓨터 시뮬레이션을 수행함으로써 제품개발 기간을 단축시키고, 설계비용도 절감하고 있다. 하지만, 차량 시뮬레이터와 같이 실시간 응답이 요구되는 경우에는 실시간 계산이 가능한 고가의 장비나 더욱 효율적인 해석 알고리즘이 요구되게 된다.

실시간 시뮬레이션을 위해서는 효율적인 차량 모델링, 시뮬레이션 시간이 짧으면서도 오차가 작은 운동방정식 풀이 방법과 적분방법의 선정 및 기타 여러가지 요소들을 고려해야 한다. 효율적인 차량모델링방법으로는 차량을 구성하는 물체중에서 물체의 중량이 차량전체의 중량에 비해

무시할 수 있을 정도로 작은 경우에는 물체로 고려하지 않고 구속조건만으로 다루어지는 무질량 링크로 모델링하는 것이 효율적이며,⁽¹⁾ 지면과의 힘전달 과정을 잘 묘사하면서도 계산시간은 빠른 타이어 모델도 필요하다.

동역학 운동방정식을 구성하는 방법에는 절대좌표계를 사용하는 방법과 상대좌표계를 사용하는 방법, 그리고 절대좌표계의 일반성과 상대좌표계의 계산 효율성을 동시에 만족해서 효율성을 높이고자 하는 속도변환법⁽²⁾ 등이 있다. ADAMS⁽³⁾와 DADS⁽⁴⁾ 프로그램은 절대좌표계를 사용하며, AutoDyn7 프로그램⁽⁵⁾은 두 방법의 장점을 혼용하는 속도변환기법을 사용한다.

구속연쇄계의 운동방정식은 사용하는 좌표계에 상관없이 각 물체의 운동방정식과 물체 사이에서 생겨나는 기구학적 구속조건을 동시에 만족하도록 미분-대수방정식으로 나타난다. 이러한 방정식을 푸는 방법으로는 구속 조건 안정화 방법과 좌표분할법이 주로 사용되고 있다. 대표적인 좌표분할방법으로 LU, QR, SVD법 등이 있다. 본 연구에서는 LU, QR 좌표분할법을 비교검토하여 차량시뮬레이션에 효율성이 높은 방법을 제시하였다.

* 회원, 부산대학교 대학원

** 회원, 부산대학교 기계공학부

E-mail : wsyoo@hyowon.pusan.ac.kr

TEL : (051)510-2328 FAX : (051)512-9835

또한 소프트웨어적으로 더 빠르게 계산하는 방법중에서 기호연산기법이 여러분야에 적용되었는데, 기호를 이용하여 운동방정식을 생성시키는 프로그램으로 SYM, MESA VERDE, NEWEUL⁽⁶⁾ 등이 있다. 따라서 절대좌표계의 일반성과 상대좌표계의 효율성을 동시에 가지는 속도변환법과 기호연산기법을 함께 적용하면 계산 효율성을 더욱 향상시킬 수 있을 것으로 기대된다.

본 연구에서는 동역학해석에 큰 영향을 주는 인자 중 하나인 좌표분할법을 비교검토하여 실시간 시뮬레이션에 적합한 좌표분할법을 선정하였다. 또한, 기호연산기법을 이용하여 FORTRAN 부함수로 계산할 부분에 기호연산기법을 적용시켜서 효율성을 확인한 뒤, 이를 속도변환법을 채용하고 있는 기존의 AutoDyn7 프로그램에 삽입하여 차량의 실시간 시뮬레이션을 달성하고자 한다.

2. 시스템의 운동방정식

2.1 차량의 모델링

본 연구에서 사용하는 차량은 전륜은 맥퍼슨 현가장치이고, 후륜은 트위스트 액슬을 장착하고 있다. 전륜의 맥퍼슨 현가장치에서 좌우 로우암을 회전-구면 무질량링크(revolute-spherical massless link)로, 좌우 타이 로드(tie rod)와 래트럴 로드(lateral rod)를 구면-구면 무질량링크(spherical-spherical massless link)로 교체하고 후륜의 트위스트 액슬에 상하 2개의 물체로 모델링된 좌우 댐퍼를 힘요소로만 고려함으로써 기존의 23개의 물체로 구성된 차량시스템을 Fig. 1과 같이 14개의 물체로 모델링 하였다.

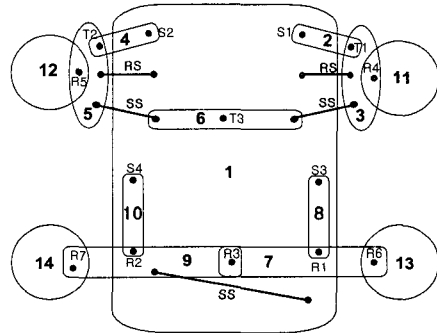
본 차량시스템은 14자유도이나 사시와 병진조인트로 연결된 조향랙(steering rack)이 변위구속조건에 의해 조향이 가능하도록 만들었기 때문에 사실상 15자유도를 가진다. Fig. 1은 차량의 조인트 연결상태를 나타내고, Table 1는 Fig. 1에 나타난 물체들의 이름들을 설명한다.

무질량 링크를 사용하여 14개의 물체로 모델링한 CAE-TC모델과 23개 물체로 모델링한 기존 모델과 비교하기 위하여 직진주행, 장애물통과, J-선회, 차선변경 시뮬레이션을 수행하였다.

Fig. 2 ~ Fig. 4는 23개의 물체로만 이루어진 차량과 무질량링크로 모델링된 CAE-TC 차량모델

에 대한 동적응답을 비교한 그림이다. 장애물 통과시는 차량속도를 20 km/h로, J-선회와 차선변경시는 50 km/h로 시뮬레이션 하였다.

모든 경우 응답이 잘 일치함을 알 수 있다. 모두 10초 동안 시뮬레이션 한 경우로서 장애물 통과시 전방 15 m에서 높이 0.1 m, 길이 2.5 m의 반사인곡선의 장애물을 만난다. 그리고 J-선



R# : Revolute joint T# : Translational joint
S# : Spherical Joint U# : Universal Joint

Fig. 1 Modeling of CAE-TC vehicle

Table 1 Body names illustrated in Fig. 1

No.	Body name	No.	Body name
1	chassis	8, 10	trailing arm
2, 4	upper strut	11, 12	front wheel
3, 5	knuckle	13, 14	rear wheel
6	steering rack	SS	spherical-spherical massless link
7, 9	rear axle	RS	revolute-spherical massless link

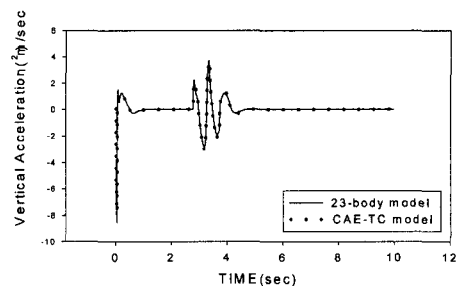


Fig. 2 Vertical acceleration of chassis(bump simulation)

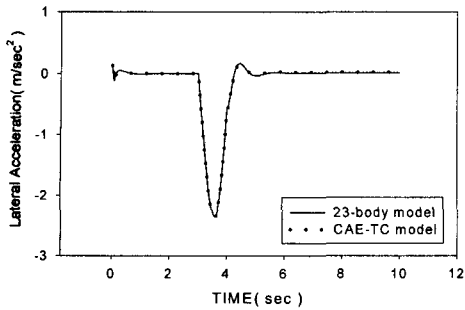


Fig. 3 Lateral acceleration of chassis(J-turn simulation)

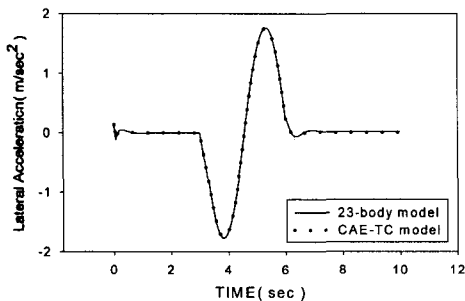


Fig. 4 Lateral acceleration of chassis (lane change simulation)

Table 2 Efficiency of the 14-body Model vs 23-body model

	Straight driving	Bump	J-turn	Lane change
Average(%)	54.8	48.6	52.7	53.2

회의 경우 랙(rack)에 1초 동안 반사인곡선을 변위로 주었으며, 차선변경인 경우에는 랙에 3초동안 사인파형의 변위를 주었다.

23개 물체 모델에 대한 14개 물체 모델의 시뮬레이션 시간효율을 Table 2에 나타내었다. 14개 물체 모델의 시뮬레이션 시간이 평균 52.3% 감소되었다.

2.2 속도변환법을 이용한 운동방정식

속도변환행렬 B를 이용하면 시스템의 운동방

정식은 다음식과 같이 바뀐다.⁽²⁾

$$\begin{aligned}
 \dot{y} &= B\dot{q} \\
 \ddot{y} &= \dot{B}\dot{q} + B\ddot{q} \\
 M(B\ddot{q} + \dot{B}\dot{q}) + \Phi_y^T \lambda &= F - S \quad (1) \\
 B^T M B \ddot{q} + B^T \Phi_y^T \lambda &= B^T (F - S - M \dot{B} \dot{q}) \\
 \overline{M} \ddot{q} + \overline{\Phi}_q^T \lambda &= B^T (F - S - M \dot{B} \dot{q})
 \end{aligned}$$

최종적으로 구속조건이 있는 시스템의 운동방정식을 미분-대수방정식(Differential-Algebraic Equation)형태로 나타내면 다음과 같다.

$$\begin{bmatrix} \overline{M} & \overline{\Phi}_q^T \\ \overline{\Phi}_q & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix} = \begin{bmatrix} \overline{Q} \\ a \end{bmatrix} \quad (2)$$

여기서 \overline{M} 는 일반질량행렬(generalized mass matrix)이고, \overline{Q} 는 수정일반력(modified generalized force)이다. 각 항을 다시 쓰면 다음과 같다.

$$\begin{aligned}
 \overline{M} &= B^T M B \\
 \overline{Q} &= B^T (F - S - M \dot{B} \dot{q}) \quad (3) \\
 a &= -(\dot{\Phi}_q \dot{q} + \dot{\Phi}_t)
 \end{aligned}$$

여기서 아래첨자 q와 t는 각각 일반 좌표와 시간에 대한 편미분을 나타낸다.

3. 대수-미분방정식의 해법

3.1 LU분해법 이용

구속조건의 자코비안이 완전 행차수(full row rank)를 가지면, 자코비안을 LU분리(LU decomposition)를 할 수 있다. 행렬 L과 U는 종속좌표 u를 R은 독립좌표 v를 나타내며 자코비안은 다음과 같은 형태로 최종 분할될 수 있다.

$$\Phi_q = [\Phi_u \Phi_v] \quad (4)$$

여기서 $\Phi_u = LU$ 이며 $\Phi_v = LR$ 이다.

위치해석시 식 (5)를 이용해서 종속좌표성분을 구한다.

$$\begin{bmatrix} \Phi_u & \Phi_v \\ 0 & I \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} -\Phi \\ 0 \end{bmatrix} \quad (5)$$

$$u^{i+1} = u^i + \Delta u$$

여기서, i 는 반복횟수를 나타낸다.

속도해석시 식 (6)을 이용하여 종속 속도벡터 \dot{u} 를 계산할 수 있다.

$$\begin{bmatrix} \Phi_u & \Phi_v \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} -\Phi_t \\ \dot{v} \end{bmatrix} \quad (6)$$

가속도 관계식을 전체 시스템의 운동방정식과 연립하여 미분 대수 방정식 형태로 최종 가속도식을 구성하면 식 (7)과 같다.

$$\begin{bmatrix} M & \Phi_q^T \\ \Phi_q & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ \lambda \end{bmatrix} = \begin{bmatrix} Q \\ -2\Phi_{qt}\dot{q} - (\Phi_q\dot{q})_q - \Phi_{tt} \end{bmatrix} \quad (7)$$

여기서 M 은 시스템 전체의 질량행렬이고, λ 는 라그랑지 곱수를 나타내며, Q 는 외력항을 나타낸다.

3.2 QR분해법 이용

일반적으로 완전 행차수(row rank)를 가지는 자코비안 행렬은 수직행렬 Q 와 상위삼각행렬 R 로 다음과 같이 분할할 수 있다.

$$\Phi_q^T = QR \quad (8)$$

또한 수직행렬을 다음과 같이 $Q = [Q_1 \ Q_2]$ 로 분할된다. 속도 관계식은 식 (9)와 같이 표현된다.

$$\dot{q} = \begin{bmatrix} \widehat{Q}_1 & \widehat{Q}_2 \end{bmatrix} \begin{bmatrix} b \\ \dot{z} \end{bmatrix} \quad (9)$$

여기서 $b = -\Phi_t(q, t)$, $\widehat{Q}_1 = Q_1(\Phi_q Q_1)^{-1}$, $\widehat{Q}_2 = Q_2 - \widehat{Q}_1 \Phi_q Q_2$ 이고 \dot{z} 는 구속조건의 접평면 성분과 평행한 자유변수벡터이다. 가속도는 식 (10), (11)로부터 구해진다.

$$\ddot{q} = \begin{bmatrix} \widehat{Q}_1 & \widehat{Q}_2 \end{bmatrix} \begin{bmatrix} a \\ \ddot{z} \end{bmatrix} \quad (10)$$

$$\ddot{z} = M_{11}^{-1} \widehat{Q}_2^T (g - M \widehat{Q}_1 a) \quad (11)$$

여기서 \ddot{z} 는 계의 독립가속도 벡터이며, 크기는 계의 자유도수와 같다. 또한,

$$M_{11} = \widehat{Q}_2^T M \widehat{Q}_2, \quad a = -\Phi_q \ddot{q} - \Phi_{tt}$$

이며, 식 (7)의 라그랑지 곱수는 다음의 식으로부터 구해진다.

$$\lambda = \widehat{Q}_1^T (M \ddot{q} - \bar{g}) \quad (12)$$

3.3 LU- QR 분해법 비교

Table 3에는 LU 좌표분할법을 사용한 경우에 대한 QR 좌표분할법을 사용한 경우의 계산효율을 나타내고 있다. 14개 물체로 모델링한 차량의 시뮬레이션에서는 QR분해법을 사용하는 경우가 평균 40.6%의 시뮬레이션 시간 감소를 가져오므로 더 효율적임을 알 수 있다. 왜냐하면 LU 분해법은 속도해석과 가속도해석시에 정방행렬을 만들어서 선형시스템을 풀게 되므로 이 정방행렬의 크기가 커질수록 계산효율은 떨어지게 된다.

CAE- TC 모델에 대해 LU분해법 사용자 풀어야 할 정방행렬의 크기는 위치 및 속도해석시에 (25 × 25)이고 가속도해석시는 (36 × 36)이다. 반면에 QR분해법을 사용하는 경우엔 위치해석시 수행되는 QR좌표분할 자체가 시간이 많이 소요되는 반면에 속도해석시에는 선형시스템을 풀지 않고 식 (9)와 같이 행렬곱으로 바로 속도가 구해지며, 가속도해석시에는 자유도수 만큼인 (14 × 14) 크기의 선형시스템을 풀게 되므로 결국 복잡한 시스템에 있어서는 QR 좌표분할법을 사용한

Table 3 Efficiency of QR decomposition vs LU decomposition(with 14-body model)

	Straight driving	Bump	J-turn	Lane change
Average(%)	36.5	47.9	37.3	40.9

는 경우가 전체 시뮬레이션 계산효율을 증가시킬 수가 있게 된다.

따라서 이후로는 모두 적절한 차량모델인 CAE-TC와 복잡한 차량시스템에 적합한 QR 좌표분할법을 사용해서 시뮬레이션을 수행한다. 그리고 적분방법으로는 다단계해법인 예측-수정자법(predictor-corrector method)인 DE⁽⁸⁾루틴을 이용하여 수치적분을 수행한다. 이 알고리즘은 Adams-Bashforth 방법을 개형다단계법으로 사용하고 Adams-Moulton 방법은 폐형다단계법으로 사용한다.

4. 운동방정식의 기호연산기법화

4.1 기호연산코드(Symbolic computation code)

일반화된 범용 프로그램의 경우에 시뮬레이터에 사용되는 특정한 차량을 대상으로 시뮬레이션하게 되면 불필요한 프로그램코드가 존재하기 마련이다. 하지만 다양한 여러 가지 모듈들을 사용할 수 있다는 장점도 있다. 따라서 차량해석 프로그램인 AutoDyn7의 여러 가지 기능들을 이용하면서 본 차량에 적합한 프로그램을 만들기 위해서 기호연산코드를 도입하였다.

일단 CAE-TC 차량시스템에 대해 수치값에 의존하는 부분을 제외한 일반 서브루틴들을 기호연산 프로그램 Mathematica⁽⁹⁾를 이용하여 기호연산을 함으로써 계산효율을 높이고자 하였다.

일반적인 수치해석 프로그램의 경우에는 입력을 받아서 필요한 출력데이터를 내보내기까지의 해석과정에서 입력력과 무관한 많은 지역변수와 배열들을 사용하여 연산을 하게 되는데 실제로 출력되는 변수는 극히 일부분이다. 따라서 정의된 시스템으로부터 알고 있는 정보를 이용하여 그 입력과 출력 사이의 계산을 Mathematica에서 기호로 미리 연산을 해서 입력과 출력에 관계된 변수만으로 구성된 FORTRAN 부함수를 출력하여 줌으로써 이를 이용하여 계산효율을 높이고자 하

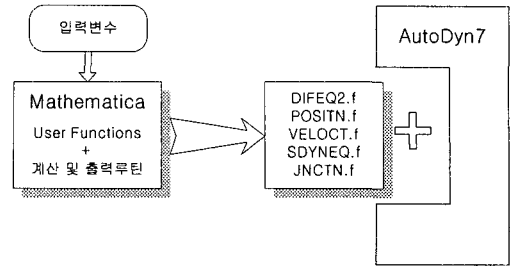


Fig. 5 Connection between Mathematica and AutoDyn7 program

Table 4 Computation time ratio according to analysis steps in AutoDyn7 (straight line driving)

	Position analysis	Velocity analysis	Acceleration analysis
Computation time(%)	36.9	14.4	47.6

는 방법이 본 연구에서 사용하는 기호연산기법이다.

Fig. 5에는 기호연산 프로그램 Mathematica로부터 출력된 FORTRAN 부함수들을 해석부분별로 묶어서 기존의 AutoDyn7과 함께 사용하는 연결을 보여주고 있다. 프로그램 실행시 본 차량시스템에 적합하게 구성된 부함수들로 계산을 하게 되므로 계산효율성을 가져올 수 있고, 또한 여러 가지 타이어 비교 시뮬레이션의 경우 간단한 입력데이터 조작만으로 AutoDyn7의 모듈들을 사용할 수 있다는 장점이 있다.

이제 각 해석단계별로 기호연산코드의 활용성을 알아보고자 한다. 우선 시뮬레이션 전반에 관하여 해석부분별 계산비중을 살펴보면 Table 4와 같다. 위치해석, 속도해석 그리고 가속도해석 부분이 전체 시뮬레이션 시간중에서 약 99%를 차지하므로, 주어진 해석시간동안에 반복계산되는 위치, 속도, 가속도해석 부분에서 해석시간을 줄이는 것이 중요하다.

4.2 위치해석의 SYMBOLIC CODE화

Table 4에서 보면 위치해석의 계산비중이 가속도해석 다음으로 큰데 이것은 QR 좌표분할과정에서 속도 및 가속도해석시 필요한 변수들을 미

Table 5 Fortran subroutines required in the position analysis

POSITN.f		
Mathematica Files	Fortran Subroutines	Contents
positn.nb	POSITNS, CHKCNVS, MTRAONE	위치해석전반에 관련된 부함수
qtox.nb	TRANQXS, QTOXS, UPDTPS, EULERS	상대좌표를 절대좌표로 변환하는데 필요한 부함수
matxb.nb	MATXBS	속도변환행렬 B 를 구성하는데 필요한 부함수
qxd.nb	QDXDS	상대속도를 절대속도로 변환하는데 필요한 부함수
matxbd.nb	MATXBDS	속도변환행렬의 시간미분인 \dot{B} 을 구성하는데 필요한 부함수

Table 6 Fortran subroutines required in the velocity analysis

VELOCT.f		
Mathematica Files	Fortran Subroutines	Contents
veloct.nb	VELOCTS, MATVEONE	속도해석전반에 관련된 사항들

리 계산해 놓기 때문이다. 따라서 QR 좌표분할 과정을 제외한 나머지 부분은 복잡한 계산루틴들은 많으나 그 계산비중이 적어서 Table 8에서 보듯이 계산시간이 12.9% 감소하였으며, 다른 해석 부분과 비교해서 감소폭이 크게 높지는 않았다.

4.3 속도해석의 SYMBOLIC CODE화

QR 좌표분할법을 사용하여 속도를 구하는 경우엔 단순히 행렬곱으로 속도항이 구해지게 되므로 전체계산시간에 대한 계산비중은 Table 4에 나타나 있는 바와 같이 크지 않다. 기호계산기법을 적용한 결과 약 28.3% 시뮬레이션 시간이 줄어들었다.

4.4 가속도해석의 SYMBOLIC CODE화

Table 7에는 Mathematica로부터 출력된 가속도 해석에 관련된 FORTRAN 부함수들을 설명한다. 운동방정식 구성요소인 일반질량행렬과 일반력 기타 계산루틴들을 기호연산기법을 이용하여 처

Table 7 Fortran subroutines required in the acceleration analysis

SDYNEQ.f		
Mathematica Files	Fortran Subroutines	Contents
sdyneq.nb	SDYNEQS	가속도해석전반에 관련된 사항들
	MASMTXS	질량행렬계산
	BTMBS	속도변환행렬을 이용하여 일반질량행렬계산
	FORCES	힘계산
	BTFS	속도변환행렬을 이용하여 일반력계산
	RHSAC2S	구속식의 2차미분항 계산

Table 8 Efficiency of the analysis steps using the symbolic computation method(straight line driving)

Part	Type	Nonsymbolic (sec)	Symbolic (sec)	Reduction Ratio(%)
Position analysis		2.42	2.11	12.9
Velocity analysis		0.94	0.67	28.3
Acceleration analysis		3.12	2.15	31.0

Table 9 Efficiency of CAE-TC applying a symbolic computation method to AutoDyn7

	Straight driving	Bump	J-turn	Lane change
Average(%)	22.4	22.8	22.5	22.0

리하였다. 계산은 위치해석에 비해 간단하나 계산량이 많기 때문에 기호연산한 결과 계산시간이 약 31% 줄어들었다.

4.5 SYMBOLIC CODE의 효율성

앞절에서 설명한 위치, 속도, 가속도 해석을 모두 기호연산하여 얻은 시뮬레이션의 효율이 Table 9에 나타나있다. 기호연산법을 사용하지 않은 경우와 비교하여 시뮬레이션 시간이 평균 22.4% 감소되었음을 확인할 수 있었다.

Table 10에는 속도변환법을 사용하는 AutoDyn7

Table 10 Simulation times of CAE-TC applying a symbolic method to AutoDyn7

Velocity (km/h)	Straight driving(sec)	Bump (sec)	J-turn (sec)	Lane change(sec)
10	5.64	6.14	6.05	6.0
20	4.38	5.41	4.99	4.78
30	4.28		4.75	4.73
40	4.57		4.94	5.0
50	4.99		5.55	5.54
60	5.09		5.79	5.89
70	5.47		6.0	6.36
80	6.0		6.53	6.8

프로그램에 기호연산법을 적용한 경우의 여러 종류의 동역학 시뮬레이션에 소요되는 컴퓨터시간을 기록하였다. SGI(Silicon Graphics Inc.) 워크스테이션 Octane(195MHz CPU 내장)을 사용한 결과, 10초 시뮬레이션을 수행하는데 필요한 시간이 7초 이내임을 확인할 수 있었다.

Table 10에서 초기속도에 따라서 시뮬레이션 시간이 차이가 나는데 이것은 초기속도에 따른 차량의 고유특성으로서 특히 장애물통과 시뮬레이션에서 두드러지게 나타난다. 말하자면 반사인 곡선형태의 장애물의 길이에 따른 장애물의 진동수와 초기속도에 따른 차량의 최고 수직가속도가 발생되는 진동수가 근접하는 경우에 공진이 발생하여 시뮬레이션이 2배이상 상당히 많이 지연되는 현상이 나타났다.

5. 결론

본 연구에서 차량모델링과 운동방정식풀이방법에 따른 시뮬레이션 시간을 비교검토해 보았고, 기호연산기법을 적용하여 계산효율을 상승시켰다.

(1) 적절한 차량모델링이 계산효율에 가장 큰 영향을 미침을 알 수 있었다.

(2) 운동방정식의 풀이방법 중에서 복잡한 차량 시스템에 적합한 방법은 QR 좌표분할법임을 확인하였다.

(3) 기호연산기법을 해석부분별로 비교검토하여 본 결과 계산효율이 상승함을 알 수 있었고, 차량 해석 프로그램인 Autodyn7에 적용한 결과 전

체 시뮬레이션 계산효율이 상당히 증가함을 알 수 있었다.

후 기

본 연구는 한국과학재단 특정연구과제(과제번호 97-0200-1001-5)의 지원에 의해 수행되었음.

참고문헌

- (1) 유완석, 김광석, 정보선, 이종년, 1999 "실시간 시뮬레이션을 위한 효율적인 차량 모델링 - 힘 요소가 고려된 무절량링크의 효과 분석," 자동차 공학회논문집, 제7권, 제4호, pp. 330~337.
- (2) Kim. S. S., Vanderploeg M. J., 1984 "A State Space Formulation for Multibody Dynamic Systems subject to Control," Univ. of Iowa, Tech. Report No. 84-20, Dec.
- (3) 1994 ADAMS(Automated Dynamic Analysis and Mechanical System) User's Manual, *Mechanical Dynamic Inc. U.S.A.*
- (4) 1995 DADS(Dynamic Analysis and Design System) User's Manual, *Computer Aided Design Software Incorporated, Iowa, U.S.A.*
- (5) 김광석, 유완석, 김성수, 김상섭, 1999 "차량동역학 해석 프로그램 AutoDyn7의 개발(I) - 프로그램 구성 및 주요 알고리즘," 자동차공학회논문집, 제7권, 제3호, pp. 321~330.
- (6) Schiehlen W., 1990, *Multibody Systems Handbook*, Springer-Verlag.
- (7) 김의조, 유완석, 1997 "준역행렬과 투영행렬을 이용한 구속다물체계의 동역학 해석," 대한기계학회논문집, 제21권, 제8호, pp. 1311~1321.
- (8) Shampine, L. F. and Gordon, M. K., 1975, "Computer Solution of Ordinary Differential Equations : The Initial Value Problem," W. J. Freeman, San Francisco, California.
- (9) Wolfram, S., 1991, *Mathematica*, Addison Wesley.
- (10) 이운성, 이지선, 김상섭, 1994 "실시간 차량 시뮬레이션을 위한 심불운동방정식의 생성," 한국자동차공학회 추계학술대회논문집, pp. 327~331.