Trans. KIEE, Vol. 37, No. 4, APR. 1988

論 文
37~4~5

개인용 컴퓨터를 이용한 로보트 매니퓨레이터의 동적 방정식의 자동 생성에 관한 연구

Automatic Generation of Dynamic Equations for Robotic Manipulators using Personal Computer

黃 祖 善*·崔 榮 奎**·元 太 鉉***·徐 正 一***
(Chang-Sun Hwang · Young-Kiu Choi · Tae-Hyun Weon · Jeong-Il Seo)

요 약

본 연구에서는, 기호연산이 가능한 muSIMP / MATH package를 이용하여 로보트 매니퓨레이터의 동적 방적식을 구할 수 있는 프로그램을 개발하였다. LISP를 기초로 하여 만들어진 컴퓨터 대수 package인 muSIMP / MATH는 수, 변수, 함수 그리고 행렬을 포함한 대수학 수식 조작이 가능하며, IBM-PC와 같은 개인용 컴퓨터에서 운용이 가능하다. 개발된 프로그램은 Lagrange-Euler방법에 의거하여 구성되었으며, 자유도가 6 이하인 임의의 로보트 매니퓨레이터에 대한 적용이 가능하다.

Abstract- A program is developed for generating the dynamic equations for the robotic manipulators using the symbolic language muSIMP / MATH. The muSIMP / MATH is a LISP-based computer algebra package, devoted to the manipulation of algebraic expressions including numbers, variables, functions, and matrices. The muSIMP-MATH can operate on personal computer such as IBM-PC. The program is developed, based on the Lagrange-Euler formulation. This program is applicable to the manipulators with any number of degrees of freedom, and maximum number of degrees of freedom is set to be six in this program.

1. 서 론

공장 자동화의 필요성이 강조됨에 따라 로보트의 중요성이 크게 부각되어지고 있다. 이에 따라 로보트를 제어하기 위한 여러방법이 제시되고 있으며, 힘 제어 혹은 위치 제어를 위해서 로보트 매니퓨레이터의 운동학(kinematics)과 동력학(dynamics) 의 해석이 필수적인 요건으로 부각되고 있다.

로보트 매니퓨레이터의 동력학을 해석하기 위한 공식으로는 라그랑지-오일러(Lagrange-Euler)[1),2),3)] 방법과 뉴톤-오일러(Newton-Euler)[2),4)] 방법 등이 알려져 있다. 이들 두 가지 방법을 비교하면 뉴톤-오일러 방법이 계산상의 효율성은 좋은 반면에, 라그랑지-오일러 방법은 수식의 형태가 간단하다. 여기서 계산이 효율적이라는 것은 동적 방정식이 벡터/행렬로 나타내어 진다는 전제에 기초를 두며 관절력(혹은 토오크)를 구하는데 수치연산을 사용한다는 점에서의 비교이다.[5),6)] 만일 벡터/행렬 방정식을 기호연산[7)] (symbolic computation)에 의해 스칼라 형태로 풀어 쓴다면, 뉴톤-오일러 방법이나 라그랑지-오일러 방법의 결과 방정식은 같아진다.[5)] 그러나 동적 방정식의 중간과정과 각각의 항을 검토하기 위해서는 폐형식(closed form)의 방정식이 요구되므로 라그랑지-오일러 방법을 사용하는 것

*正 會 員 : 釜山大 工大 電氣工學科 敎授 · 工博
**正 會 員 : 釜山大 工大 電氣工學科 助敎授 · 工博
***正 會 員 : 釜山大 大學院 電氣工學科
接受日字 : 1987年 11月 25日
1次修正 : 1988年 2月 8日

이 편리하다.

동적 방정식을 이용하여 로보트 매니퓨레이터를 제어함에 있어서 계산 효율을 높이기 위해서는 제어용 프로세서의 연산과정을 간단하게 처리되도록 하기 위하여 유도된 결과 방정식이 단순화된 기호 형태로 나타내져야 하므로 기호연산이 필요하게 된다.

기호연산은 일반적으로 수작업이 요구되므로 시간이 오래 걸리며 오류가 발생할 확률이 높다. 또한 유도과정에는 수많은 벡터와 행렬연산이 필요하며, 유도된 방정식도 항이 많아서 구조가 단순한 로보트의 경우에 있어서도 수작업은 곤란한 형편이다. 이러한 문제점을 해결하기 위하여 컴퓨터를 이용할 필요성이 있다.[8]

기호연산을 행하기 위한 컴퓨터 언어는 일반적으로 인공 지능 언어이며, 이러한 언어로써 로보트 매니퓨레이터의 동적 방정식을 유도한 논문들은 LISP나 PROLOG 언어 및 LISP를 토대로 하여 만든 algebra package인 MACSYMA[6], REDUCE[9], ARM[10] 등을 이용하였다. 그러나 이들 package들은 대형 혹은 중형 컴퓨터에서 운용되므로 이러한 package들이 운용되는 컴퓨터가 없을 경우 사용상의 문제가 있으므로 이러한 package와 동일한 기능이 개인용 컴퓨터에서 행하여 진다면 바람직할 것이다. 개인용 컴퓨터에서 실행 가능한 PROLOG[11]나 LISP[12] 언어를 이용하여 동적 방정식을 구하는 논문도 소개되어 있으나, 이들 언어를 사용하려면 동적 방정식을 유도하는데 필요한 모든 함수를 만들어야 하므로 프로그램이 커져서, 보통의 개인용 컴퓨터의 주기억장치 용량으로는 감당하기 힘든 문제점이 있다.

본 논문에서는 이러한 문제점을 개선하기 위하여 기호연산 기능이 있는 muSIMP/MATH[13]를 사용하여 IBM-PC 호환기종(OS는 MS-DOS, 주기억장치는 640KBytes)에서 로보트 매니퓨레이터의 동적 방정식을 라그랑지-오일러 방법을 기초로 하여 자유도가 6 이하인 임의의 자유도를 가진 로보트 매니퓨레이터의 경우에 대하여 관절의 종류에 관계없이 구할 수 있는 소프트 웨어를 개발한다.

2. 라그랑지-오일러(Lagrange-Euler) 방정식

로보트의 운동학은 데나비트-할텐버그 표시법과 동차 변환(homogeneous transformation) 방법을 이용하여 표현할 수 있다. 데비나트-할덴버그 방법은 관절로 된 각 링크에 좌표계를 설정하여 전체계를 해석하는 행렬해석 방법이다. 이것은 앞의 링크 좌표계에 대한 다음 링크좌표계의 관계를 링크의 기하학적 파라메터 α_i, θ_i, a_i, d_i가 포함된 (4×4) 동차 변환 행렬(homogeneous transformation matrix)로써 결정한다.

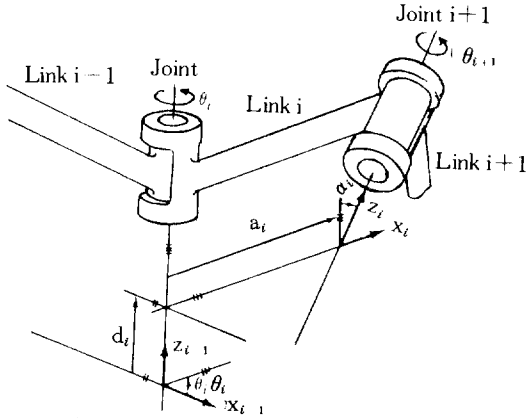

그림 1. 링크 파라메터 α_i, θ_i, a_i, d_i

Fig. 1. Link parameter α_i, θ_i, a_i, d_i

그림 1에서 링크 i−1 좌표계와 링크 i의 좌표계 사이의 상호 관계인 (4×4)동차 변환 행렬 ${}^{i-1}A_i$를 구하면 다음과 같다.

$$ {}^{i-1}A_i = \mathrm{Rot}(Z_{i-1}, \theta_i)\ \mathrm{Trans}(0, 0, d_i)\ \mathrm{Trans}(a_i, 0, 0)\ \mathrm{Rot}(X_i, \alpha_i) $$

$$ = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.1) $$

로보트의 관절 종류에 따라서, 회전 관절인 경우 θ_i가, 미끄럼 관절인 경우 d_i가 변수로 된다. 링크 j−1 좌표계로 부터 링크 i좌표계의 전향 변환행렬은 다음과 같다.

$$ {}^{j-1}T_i = {}^{j-1}A_j\ {}^{j}A_{j+1} \cdots {}^{i-1}A_i \qquad (2.2) $$

라그랑지안(Lagrangian) L은 시스템의 운동에너지 K와 위치에너지 P와의 차로 정의되며 다음과 같다.

$$ L = K - P \qquad (2.3) $$

라그랑지-오일러 방정식의 일반형태는 다음과 같다.

$$F_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{Q}_i}\right) - \frac{\partial L}{\partial Q_i} \qquad (2.4)$$

여기에서 Q_i 는 관절 변수

$\dot{Q}_i$ 는 Q_i의 시간에 대한 미분치

F_i는 시스템에 인가된 힘(혹은 토오크)

이를 n자유도를 가진 로보트 매니퓰레이터에 대하여 적용하면 다음과 같은 동적 방정식을 구할 수 있다.

$$F_i = \sum_{k=1}^{n} D_{ik}\ddot{Q}_k + Ia_i\ddot{Q}_i + \sum_{j=1}^{n}\sum_{k=1}^{n} H_{ijk}\dot{Q}_j\dot{Q}_k + G_i \qquad (2.5)$$

여기서,

1) F_i는 관절 i에 적용되는 일반화된 힘(혹은토오크)이다.

2) D_{ik}는 관성 행렬로서 식(2.6)와 같이 나타낼 수 있으며, 대칭이고 양의 정치(positive definite)로서 역이 항상 존재한다.

$$D_{ik} = \sum_{p=\max(i,j,k)}^{n} T_r\left(\frac{\partial T_p}{\partial Q_k} J_p \frac{\partial T_p^T}{\partial Q_i}\right) \qquad (2.6)$$

3) H_{ijk}는 관절 속도에 영향을 받는 항으로서 원심력과 코리올리스 힘의 합이며 다음과 같이 표현된다.

$$H_{ijk} = \sum_{p=\max(i,j,k)}^{n} T_r\left[\frac{\partial^2 T_p}{\partial Q_j \partial Q_k} J_p \frac{\partial T_p^T}{\partial Q_i}\right] \qquad (2.7)$$

4) G_i는 중력 부하력으로서 중력 상수 G가 포함된 모든 항으로 다음과 같이 쓸 수 있다.

$$G_i = \sum_{p=i}^{n} -M_p G_r^T \frac{\partial T_p}{\partial Q_i} {}^p r_p \qquad (2.8)$$

5) J_i는 링크 i의 링크 i좌표계에 관한 의사 관성 행렬로서 다음과 같이 나타낼 수 있다.

$$J_i = \begin{bmatrix} \frac{-I_i^{xx}+I_i^{yy}+I_i^{zz}}{2} & I_i^{xy} & I_i^{xz} & M_iX_i \\ I_i^{xy} & \frac{I_i^{xx}-I_i^{yy}+I_i^{zz}}{2} & I_i^{yz} & M_iY_i \\ I_i^{xz} & I_i^{yz} & \frac{I_i^{xx}+I_i^{yy}-I_i^{zz}}{2} & M_iZ_i \\ M_iX_i & M_iY_i & M_iZ_i & M_i \end{bmatrix} \qquad (2.9)$$

6) 벡터 ${}^i r_i$는 링크 i의 질량 중심의 위치를 나타내며, 다음과 같이 표현할 수 있다.

$${}^i r_i = [X_i \; Y_i \; Z_i \; 1]^T \qquad (2.10)$$

7) 벡터 G_r는 중력장을 나타내며, 다음과 같이 표현할 수 있다.

$$G_r = [G_x \; G_y \; G_z \; 0]^T \qquad (2.11)$$

8) I_{a_i}는 각 관절에 있는 액츄에이터(actuator)의 관성 모멘트를 나타낸다.

3. muSIMP/MATH를 기초로 한 라그랑지-오일러 방정식의 기호연산

muSIMP/MATH은 기호연산 기능을 가지고 있으므로 이를 이용하여 동적 방정식을 구하는 프로그램을 개발할 수 있다. 표1은 muSIMP/MATH의 이러한 기능을 도식적으로 나타내고 있다.

표 1. muSIMP/MATH의 기호연산

Table 1. Symbolic computation of muSIMP/MATH

expression of muSIMP/MATH	result
(SIN θ)²+(COS θ)²	1
1 * θ	θ
θ^4 * θ^(1/2)	θ^(9/2)
θ * θ	θ^2

LISP나 PROLOG 언어로 동적 방정식을 유도할 경우 기호 연산이 가능하도록 하는 모든 함수(사칙연산과 수식 단순화가 가능한 함수)들을 미리 정의해야 하나, muSIMP/MATH에서는 이러한 기능이 내장되어 있으므로 관계된 함수의 정의를 고려하지 않아도 된다.

다음은 muSIMP/MATH에서 제공되는 기호 연산자중 본 프로그램에 사용한 함수들이다.

1. 두 행렬 A, B의 곱은 아래와 같이 구하면 된다.

   A. B

2. 행렬 A의 전치행렬은 아래와 같이 구할 수있다.

   A'

3. 행렬 A를 변수 θ으로 미분하려면 다음과 같은 방법을 사용한다.

DIF(A, θ)

DIF함수는 실제로 편미분이 가능하다.

4. muSIMP/MATH에 정의되어 사용가능한 대수학적인 단순화 함수(algebra simplification function)들은 EXPD, EXPAND, FCTR, TRGEXPD 등이 있으며, 각 함수들의 기능은 표 2 와 같다.

표 2. 수식 단순화

Table 2. Simplification of expression

함 수	기 능
EXPD	다항식을 전개
EXPAND	다항식을 공통분모로 전개
FCTR	다항식을 공통인수로 전개
TRGEXPD	삼각함수 다항식을 전개

이를 기초로 하여 그림 2 와 같은 순서도를 가지는 프로그램을 개발할 수 있다.

로보트 매니퓨레이터의 자유도를 먼저 입력하고 관절 변수를 입력하면 관절의 종류에 따라서 3 개의 링크 파라메터의 입력을 요구하게 된다. 이때에 운동전향 행렬이 동시에 구해진다. 관성 행렬의 요소 6 개와 링크의 질량 중심 및 중력 벡터를 입력하면 step 1 은 종료된다.

step 2 는 전향 변환 행렬을 구하는 과정으로서 행렬의 합 및 곱만의 절차(procedure)로 구성되어 있다.

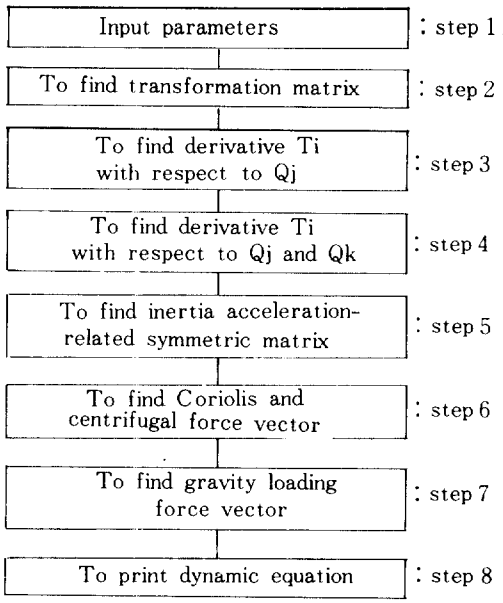

그림 2. 프로그램의 순서도

Fig. 2. Flowchart of program

step 3 과 4 는 전향 변환 행렬을 각 관절 변수로서 1, 2 차 미분을 만들어 내는 절차이다.

step 5 는 step 1 과 step 3 로 부터 의사 관성 행렬과 전향 변환 행렬의 1 차 미분 결과를 받아서 관성 행렬 D_{ik}를 구성한다.

step 6 은 step 1 과 step 3, 4 로부터 의사 관

표 3. 중력 벡터를 구하기 위한 프로그램 리스트

Table 3. Program list to find gravity loading force vector

```
FUNCTION GI(numI, numP)
numI : 0, %기호 :는 FORTRAN의 할당문(=)과 동일                    %
LOOP    %외부 loop의 시작                                        %
    numI : numI+1,
    WHEN numI>dof EXIT, %외부 loop의 탈출 조건                   %
    numP : numI-1,
    Sum1 : 0,
    LOOP  %내부 loop의 시작                                      %
        numP : numP+1,
        WHEN numP>dof EXIT, % 내부 loop탈출 조건                 %
        Sum1 : Sum1-MASS[numP]. GRA'. UIJ[numP, numI].
                Position[numP],
    ENDLOOP % 내부 loop의 끝                                     %
    Grav[numI] : FCTR(TRGEXPD(Sum1)),
ENDLOOP % 외부 loop의 끝                                         %
ENDFUN$
```

성 행렬과 전향 변환 행렬의 2차 미분 결과를 받아서 원심력과 코리올리스 힘 H_{ijk}를 구성한다.

step 7은 step 1과 step 3으로 부터 각 링크의 질량, 질량 중심 및 중력 가속도와 전향 변환행렬을 받아서 중력 부하력인 G_i를 유도한다.

step 8은 step 5와 6, 7로 부터 로보트 매니퓨레이터의 동적 방정식을 유도한 결과를 출력한다.

각각의 행렬 혹은 벡터값의 출력이 요구될 경우에는 필요한 절차까지 수행시킨 후 출력시키면 되며, 전체 프로그램을 수행했을 경우에는 재실행없이 구할 수 있다.

step 7의 실제 프로그램을 소개하면 표 3과 같다.

GI라는 함수는 local 변수로 numI, numP를 사용하고, dof는 로보트 매니퓨레이터의 자유도를 의미하며, MASS[numP]는 link numP의 질량을 나타낸다. 또한 GRA는 중력장을 표현하며, Position[numP]는 link numP의 질량 중심을 의미하는데, dof와 MASS와 GRA및 Position은 그림 2의 Flow Chart 중 step 1에서 입력된다.

UIJ[numP, numI]는 전향 변환 행렬 T[numP]의 numI에 해당되는 관절변수에 대한 1차 미분으로 step 3에서 구해진다. 내부 loop Sum1은 식(2.8)을 계산하기 위하여 프로그램한 것이다. 외부 loop는 자유도보다 큰 link가 존재하지 않으므로 이를 처리하기 위한 것이다. GRAV[numI]에 식(2.8)에서 G_i에 해당하는 결과 방정식이 기억된다.

4. 적용 사례

스탠포드 매니퓨레이터(Stanford manipulator)는 그림 3과 같이 5개의 회전 관절과 1개의 미끄럼 관절을 가진 자유도가 6인 로보트이다. 본 논문에서는 손목의 회전 관절 3개가 고정된 즉 자유도가 3인 경우의 동적 방정식을 유도한다.

입력자료는

1) 자유도 : 3
2) 관절 변수 : R-R-P
3) 데나비트-할덴버그 파라메터는 표 4와 같다.
4) 의사 관성 행렬의 파라메터

I_i^{xx}, I_i^{yy}, I_i^{zz}
$I_i^{xy}=I_i^{xz}=I_i^{yz}=0$
for i= 1, 2, 3

5) 질량 및 질량중심

M_1, M_2, M_3
${}^1r_1=[\,0\ \ Y_1\ Z_1\ \ 1\,]^T$

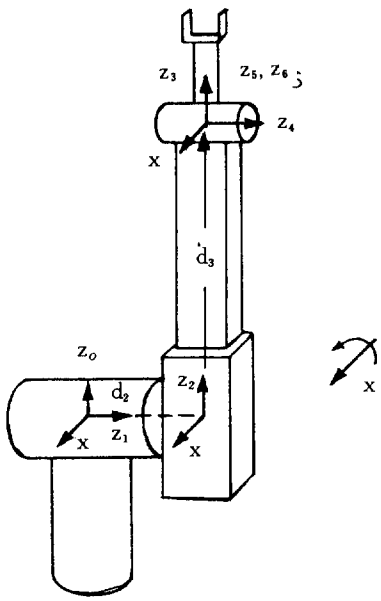

그림 3. 스탠포드 매니퓨레이터
Fig. 3. Stanford manipulator

${}^2r_2=[\,0\ \ Y_2\ \ 0\ \ 1\,]^T$
${}^3r_3=[\,0\ \ 0\ \ Z_3\ \ 1\,]^T$

6) 중력장

$G_r=[\,0\ \ 0\ \ -G\ \ 0\,]^T$

표 4. 3자유도 로보트의 데나비트-할덴버그 파라메터
Table 4. Denavit-Hartenburg parameter of 3 link robot

link No.	θ_i	α_i	d_i	a_i	변 수
1	θ_1	−90°	0	0	$Q=\theta_1$
2	θ_2	90°	D_2	0	$Q=\theta_2$
3	0°	0°	D_3	0	$Q=D_3$

위의 파라메터를 입력하여 프로그램을 수행시키면 아래와 같은 결과가 출력된다.

```
F1= (IYY1+IZZ2+IZZ3+M2 D2*(D2+2 Y2)
    +M3*(Q3*(Q3+2 Z3)*(SIN Q2)^2+
    D2^2)+(IXX2+IXX3-(IZZ2+IZZ3))*
    (SIN Q2)^2+Ia1)*ddQ1
    +(-D2 M3*(Q3+Z3)COS Q2)*ddQ2
    +(-D2 M3 SIN Q2)*ddQ3
    +(2 Q3 M3 Z3 SIN(2 Q2)+IXX2 SIN(2
      Q2)-IZZ2 SIN(2 Q2)
      +IXX3 SIN(2 Q2)-IZZ3 SIN(2 Q2)
      +Q3^2 M3 SIN(2 Q2))*dQ2*dQ1
```

```
      + (D2 M3 * (Q3 + Z3) SIN Q2) * dQ2 ^ 2
      + (Q3 M3 - Q3 M3 COS (2 Q2) + M3 Z3
        - M3 Z3 COS (2 Q2)) * dQ3 * dQ1
      + (- 2 D2 M3 COS Q2) * dQ3 * dQ2
F2 = (-D2 M3 * (Q3 + Z3) COS Q2) * ddQ1
      + (IYY2 + IYY3 + Q3 M3 * (Q3 + 2 Z3) +
        Ia2) * ddQ2
      + ((IZZ2 + IZZ3 - (IXX2 + IXX3 + Q3 M3 *
        (Q3 + 2 Z3))) * SIN (2 Q2) /2) * dQ1 ^ 2
      + (2 Q3 M3 + 2 M3 Z3) * dQ3 * dQ2
      + M3 - G * (Q3 + Z3) SIN Q2
F3 = (-D2 M3 SIN Q2) * ddQ1
      + (M3 + Ia3) * ddQ3
      + (- M3 (Q3 + Z3) (SIN Q2) ^ 2) * dQ1 ^ 2
      + (- M3 (Q3 + Z3)) * dQ2 ^ 2
      + M3 G COS Q2
```

5. 계산시 고찰사항

동적 방정식을 유도할 경우에는 많은 기호연산자가 요구된다. 특히 미분의 경우 자유도가 n인 매니퓨레이터의 경우 $\partial T_i / \partial Q_j$ matrix에 대하여 $n(n+1)/2$번의 미분계산이, $\partial^2 T_i / \partial Q_j \partial Q_k$에 대하여 $n(n+1)(n+2)/6$번의 미분이 요구되므로 자유도가 높으면 높을수록 그 계산량은 기하 급수적으로 증가하게 되어 컴퓨터 주 기억장치가 부족하게 되는 경우가 생긴다. 따라서 본 프로그램은 동적 방정식의 출력항외의 모든 변수를 가능한 한 local 변수로 처리하여 자유도가 6 이하의 경우에는 IBM-PC 호환 기종(주 기억 장치 640K 바이트)에서 처리가능하도록 하였다. 계산속도는 CPU가 80286이고 clock이 8 MHz인 IBM-PC AT호환기종의 경우 모든 변수를 global 변수로 하였을 때는 35초, local 변수로 하였을 때는 41초로 약 6 초 정도가 늦어지는 것을 확인할 수 있었다. 계산속도를 증가시키기 위하여 D_{ik}와 H_{ijk}항 중 대칭인 것은 다음의 특성에 의하여 계산과정을 축소시켰다.

$$H_{iii} = 0$$
$$H_{iik} = 0 \quad \text{if } i > k$$
$$H_{ijk} = -H_{kji} \quad \text{if } j < i, k$$
$$D_{ik} = D_{ki}$$

6. 결 론

본 논문에서는 개인용 컴퓨터에서 기호 연산이 가능한 muSIMP/MATH를 이용하고 라그랑지-오일러 방법을 기초로 하여 로보트 매니퓨레이터의 동적 방정식을 자동 생성하는 소프트 웨어를 개발하였다.

본 연구는 한국과학재단의 지원으로 이루어졌기에 이에 감사드립니다.

참 고 문 헌

1) Paul, R., "Robot manipulators : Mathmatics, Programming, and Control", M.I.T. Press, 1981
2) Walker, M. W., and Orin, D. E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms", ASME Journal of Dynamic Systems, Measurement, and Control, Vol. 104, Sept. 1982, pp. 205-211
3) Hollerbach, J. M., "A Recursive Formulation of Lagrangian Manipulator Dynamics", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-10, No. 11, 1980, pp. 730-736
4) Luh, J.Y.S., Walker, M.W., and Paul, R.P., "On-Line Computation scheme for Mechanical Manipulators", ASME Journal of Dynamic Systems, Measurement, and Control, Vol. 102, June 1980, pp. 69-76
5) K.S. Fu, R.C. Gonzalez, and C.S.G. Lee, "ROBOTICS : Control, Sensing, Vision, and Intelligence", McGraw-Hill, 1987
6) Leu, M. C. and Hemati, N., "Automated Symbolic Derivation of Dynamic Equations of Motion for Robotic Manipulators", ASME of Dynamic Systems, Measurement, and Control, Vol. 108, Sept. 1986, pp. 172-179
7) Stoutemyer, D.R. "LISP Based Symbolic Math Systems", BYTE, Aug. 1979, pp. 176-192
8) Vecchio, L, Nicosia, S., Nicolo, F., and Lentini, D., "Automatic Generation of Dynamic Models of Manipulators", 10th International Symposium on Industrial Robotics, Milan, Italy, Mar. 1980, pp. 5-7
9) J.Y.S. Luh, and C.S. Lin, "Automatic Generation of Dynamic Equations for Mechanical Manipulators", Proceeding of JACC, June 17-19, 1981,
10) J.J. Murray, and C.P. Neuman, "ARM : An Algebraic Robot dynamic Modeling Program", Proceeding of International Conference on Robotics, 1984, pp. 103-114
11) Neto, J.L., Pereira, A.E.C., and Alves, J.B., "Symbolic Computation applied to Robot Dynamic Modeling", Proceeding of the 16th Inernational Symposium on Industrial Robots, 1986, pp. 389-400
12) Paul, R., and Izaguirre, A. "Automatic Generation of the Dynamic Equations of The Robot Manipulators using a LISP program", IEEE Conference on Robotics Automation, 1986, pp. 220-226
13) "muSIMP / MATH Reference Manual", The Soft Warehouse, Sept., 1983