

로보트 팔의 제어를 위한
Dynamics 방정식들에 관한 연구

김승배, *오세정, 최도영, 박인갑, 김형래

건국대학교 전자공학과

A Study on Dynamic motion equations
for a Robot manipulator

Seung Bai Kim, Sei Jeong Oh, Doe Yeong Choi, In Kap Park, Hyung Lae Kim
Dept. of Electronic engineering
Kon Kuk university

ABSTRACT

In this paper, it is dealt with the dynamic motion equations for a robot arm. Four kinds of the dynamic equations which are the Lagrange-Euler equations, the Recursive L-E equations, the Newton-Euler equations and the improved N-E equation are derived on robot PUMA 600. Finally the algorithms on these equations are programmed using PASCAL and are compared with each other. As the results, it is found that the improved N-E equations has the most fastest execution time among the equations and can be used in real time processing.

I. 서론

이제까지 robot에 대한 연구는 운동학(Kinematics)적인 면에서의 로보트의 구조와 제어에 중점을 둔 연구가 이루어져왔으나 최근에 와서는 보다 점밀한 제어를 위하여 역학(Dynamics)적인 면을 고려한 연구가 주를 이루고 있다.⁽¹⁾⁽²⁾

그러나 robot의 Dynamics를 고려하는 경우, 계산량이 대단히 커지기 때문에 로보트의 실시간 제어가 어려워서 이를 극복하기 위한 연구들이 대두되고 있다.⁽³⁾ 현재까지 robot 팔의 운동방정식을 해석하기 위한 주요한 두 가지 방식으로서 Lagrange-Euler equations와 Newton-Euler equations를 이용한 방식이 널리 이용되고 있다. 이를 Lagrange-Euler equations를 사용한 방식은 방정식의 각 항들의 physical meaning이 명확하기 때문에 robot의 제어시스템을 설계하기 쉬우나 계산량이 너무 많고, Newton-Euler equations를 사용한 방식은 계산량은 적은 반면에 제어시스템을 설계하기 어려운 등의 서로간의 장단점이 있다. 본 논문에서는 로보트의 내 가지 Dynamics 운동 방정식과 실행시간에 관하여 연구하였다. 특히 robot의 Dynamics를 해석하기 위하여 Lagrange-Euler equations와 Recursive Lagrange-Euler equations,⁽⁴⁾ Newton-Euler equations에 관하여 알아보았고,⁽⁵⁾ improved Newton-Euler Computational scheme⁽⁶⁾에 대하여도 연구하였다. 또 이를 program화하여 실행한 실행시간에 대하여도 비교 검토하였다.

II. Lagrange-Euler equations을 이용한 dynamic equation

robot 팔의 dynamics를 해석하는 가장 중요한 방식중의 하나가 바로 이 L-E equations이다. L-E 방식의 장점은 이 L-E equations이 robot의 dynamics의 모든 특성을 포함하고 있고, 방정식의 각 항들의 physical meaning이 분명하기 때문에 robot를 제어하기 위한 controller를 설계하기 쉬운 점이다. 하지만 계산량이 너무 많아서 실시간으로 robot를 구동시키기 어렵다.

본동시 L-E equations 다음과 같다.

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = \tau_i$$

여기에서

L = 운동 에너지(K)-위치 에너지(P)

q_i = joint 번수; rotational 시 $q_i = \theta_i$,

prismatic 시 $q_i = d_i$

\dot{q}_i = q_i 의 시간에 대한 미분치

τ_i = system에 인가된 힘 (또는 토오크)

II-1. Robot 팔의 속도

r_i 를 link i 상에 고정된 한 점이라 하면, i 번째 좌표계에 대하여 homogenous 좌표로 표시 할 수 있다.

$$r_i = (x_i, y_i, z_i, 1)^T$$

또 base 좌표계에 대한 r_i 의 속도는

$$v_i = \frac{dr_i}{dt}$$

로 표시된다. 여기서 r_i 는 base 좌표계에 대한 r_i 이다. homogeneous coordinate transformation matrix를 인 ${}^i A_1, {}^0 A_1$ 를 사용하면 r_i 를 다음과 같이 표시 할 수 있다.

$$r_i = {}^0 A_1 \cdot r_i$$

여기서 ${}^0 A_1 = {}^0 A_1 \cdot {}^1 A_2 \cdot {}^2 A_3 \cdots {}^{i-1} A_i$ 이다. base 좌표계에 대한 r_i 의 절대속도는 다음과 같이 쓸 수 있다.

$$v_i = v_i = \frac{d}{dt} [r_i] = \frac{d}{dt} [{}^0 A_1 r_i] = \left[\sum_{j=1}^i \frac{\partial {}^0 A_j}{\partial q_j} \dot{q}_j \right] r_i$$

q_i 에 대한 0A_i 의 편미분은 행렬 Q_i 를 쓰면 간단히 구해진다.

$$Q_i = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}; \text{ joint } i \text{ 가 revolute 시}$$

$$Q_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}; \text{ joint } i \text{ 가 prismatic 시}$$

또,

$$\frac{\partial {}^{i-1}A_i}{\partial q_j} = Q_i {}^{i-1}A_i$$

이므로

$$\frac{\partial {}^0A_i}{\partial q_j} = \begin{cases} {}^0A_1 {}^1A_2 \dots {}^{i-2}A_{j-1} Q_j {}^{j-1}A_j \dots {}^{i-1}A_i & ; \text{ for } j \leq i \\ 0 & ; \text{ for } j > i \end{cases}$$

위식은 link i 상의 모든 점에 대한 joint j 의 운동의 영향이라 할 수 있다.

$$U_{ij} = \frac{\partial {}^0A_i}{\partial q_j}$$

라 하면

$$U_{ij} = \begin{cases} {}^0A_{j-1} Q_j {}^{j-1}A_i & ; \text{ for } j \leq i \\ 0 & ; \text{ for } j > i \end{cases}$$

이를 이용하면 v_i 는 다음과 같다.

$$v_i = \sum_{j=1}^i U_{ij} \dot{q}_j r_i$$

또, joint i 간의 상호영향은

$$\frac{\partial U_{ij}}{\partial q_k} \triangleq U_{ijk} = \begin{cases} {}^0A_{j-1} Q_j {}^{j-1}A_{k-1} Q_k {}^{k-1}A_i & ; \text{ for } i \geq k \geq j \\ {}^0A_{k-1} Q_k {}^{k-1}A_{j-1} Q_j {}^{j-1}A_i & ; \text{ for } i \geq j \geq k \\ 0 & ; \text{ for } i < j \text{ or } i < k \end{cases}$$

이다.

II-2. Robot 팔의 운동 에너지

link i 상의 모든 점의 속도를 알면 모든 link에서의 운동 에너지를 구할 수 있다. K_i 를 base에 대한 link i 의 운동 에너지라 하면, 질량 m 을 갖는 임자의 운동 에너지는 $1/2mv^2$ 이므로, 질량이 dm 인 link i 상의 한 점의 운동 에너지 dK_i 는 다음과 같다.

$$dK_i = 1/2 (\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2) dm = 1/2 \text{Tr}(v_i v_i^\top) dm$$

여기서 "Tr"은 절방 행렬의 trace operator이다. v_i 를 뒷식에 대입하면

$$dK_i = 1/2 \text{Tr} \left[\sum_{p=1}^i U_{ip} \dot{q}_p r_i \left(\sum_{r=1}^i U_{ir} \dot{q}_r r_i \right)^\top \right] dm = 1/2 \text{Tr} \left[\sum_{p=1}^i \sum_{r=1}^i U_{ip} \left(\int r_i r_i^\top dm \right) U_{ir}^\top \dot{q}_p \dot{q}_r \right]$$

$$K_i = \int dK_i$$

$$= 1/2 \text{Tr} \left[\sum_{p=1}^i \sum_{r=1}^i U_{ip} \left(\int r_i r_i^\top dm \right) U_{ir}^\top \dot{q}_p \dot{q}_r \right]$$

이다. 괄호 안의 적분항은 i 번째 좌표계에 대한 link i 상의 모든 점의 관성이다.

$$J_i = \int r_i r_i^\top dm$$

$$= \begin{bmatrix} \int x_i^2 dm & \int x_i y_i dm & \int x_i z_i dm & \int x_i dm \\ \int x_i y_i dm & \int y_i^2 dm & \int y_i z_i dm & \int y_i dm \\ \int x_i z_i dm & \int y_i z_i dm & \int z_i^2 dm & \int z_i dm \\ \int x_i dm & \int y_i dm & \int z_i dm & \int dm \end{bmatrix}$$

그러므로 robot 팔의 전체 운동 에너지는 다음과 같다.

$$K = \sum_{i=1}^n K_i = 1/2 \sum_{i=1}^n \text{Tr} \left[\sum_{p=1}^i \sum_{r=1}^i U_{ip} J_i U_{ir}^\top \dot{q}_p \dot{q}_r \right] = 1/2 \sum_{i=1}^n \sum_{p=1}^i \sum_{r=1}^i \left(\text{Tr} (U_{ip} J_i U_{ir}^\top) \dot{q}_p \dot{q}_r \right)$$

II-3. Robot 팔의 위치 에너지

robot 팔의 총 위치 에너지를 P 라 하고, link i 의 위치 에너지를 P_i 라 하면

$$P_i = -m_i g r_i = -m_i g ({}^0A_i \bar{r}_i)$$

단, $i = 1, 2, \dots, n$ 이고, \bar{r}_i 는 base 좌표계에 대한 질량 행 벡터 $(g_x, g_y, g_z, 0)^\top$ 이다.

그러므로 모든 link의 총 위치 에너지 P 는

$$P = \sum_{i=1}^n P_i = \sum_{i=1}^n -m_i g ({}^0A_i \bar{r}_i)$$

이다.

II-4. Lagrange-Euler 운동 방정식

Lagrangian function $L = K - P$ 이므로

$$L = 1/2 \sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^i \text{Tr}(U_{ij} J_i U_{ik}^\top) \dot{q}_j \dot{q}_k + \sum_{i=1}^n m_i g U_{ii} \bar{r}_i$$

이다. 뒷식을 이용하여 robot 팔의 dynamic 운동 방정식을 얻을 수 있다.

$$\tau_i = \sum_{j=1}^i \sum_{k=1}^j \text{Tr} (U_{jk} J_j U_{ji}^\top) \ddot{q}_k + \sum_{j=1}^n \sum_{k=1}^j \sum_{m=1}^k \text{Tr} (U_{jkm} J_j U_{ji}^\top) \dot{q}_k \dot{q}_m - \sum_{j=1}^n m_j g U_{jj} \bar{r}_i$$

또는

$$\tau_i = \sum_{k=1}^n D_{ik} \ddot{q}_k + \sum_{k=1}^n \sum_{m=1}^n H_{ikm} \dot{q}_k \dot{q}_m + G_i$$

뒷식을 더 간단히 표현할 수 있다.

$$\tau = D(\theta) \ddot{\theta} + H(\theta, \dot{\theta}) + G(\theta)$$

여기서

$$G(\theta) = nX_1 \text{ gravity loading force vector.}$$

$$H(\theta, \dot{\theta}) = nX_1 \text{ nonlinear Coriolis and Centrifugal force vector.}$$

$$D(\theta) = nX_n \text{ inertial acceleration-related symmetric matrix}$$

$$\tau = nX_1 \text{ applied torque/force to the joint actuators.}$$

III. Recursive Lagrange-Euler equations을 이용한 dynamic equation.

이 방식은 robot 팔의 dynamics를 보다 효율적으로 하기 위하여 속도, 가속도, 힘의 recursive한 관계를 이용하여 L-E equations를 다시 재구성한 것이다.

III-1. backward recursive Lagrangian dynamics

backward recursion은 robot의 base에서 end effector에 이르기까지의 속도와 가속도를 구하는 작업이다. Lagrangian dynamics에서 구한 힘을 recurrence한 관계를 이용하면 다음과 같이 간단하게 나타낼 수 있다.

$$F_i = \sum_{j=1}^n [\operatorname{tr}(\frac{\partial W_j}{\partial q_i} J_i \ddot{W}_j) - m_j g^T \frac{\partial W_j}{\partial q_i} \dot{r}_j]$$

여기서 W_j 는 coriolis와 centrifugal, reaction 성분으로서 다음과 같이 나타낸다.

$$\ddot{W}_j = \sum_{k=1}^i \frac{\partial W_j}{\partial q_k} \ddot{q}_k + \sum_{k=1}^i \sum_{l=i}^j \frac{\partial^2 W_j}{\partial q_k \partial q_l} \ddot{q}_k \ddot{q}_l$$

W_j 와 \dot{W}_j 의 recurrence 관계식은 W_j 를 바로 미분하여 구할 수 있다.

$$W_j = W_{j-1} A_j$$

$$\dot{W}_j = \dot{W}_{j-1} A_j + W_{j-1} \dot{A}_j = \dot{W}_{j-1} A_j + W_{j-1} \frac{\partial A_j}{\partial q_j} \dot{q}_j$$

$$\ddot{W}_j = \ddot{W}_{j-1} A_j + 2\dot{W}_{j-1} \frac{\partial A_j}{\partial q_j} \dot{q}_j + W_{j-1} \frac{\partial^2 A_j}{\partial q_j^2} \ddot{q}_j$$

$$+ W_{j-1} \frac{\partial A_j}{\partial q_j} \ddot{q}_j$$

III-2. forward recursive Lagrangian dynamics

forward recursion은 robot의 end effector에서 이르기까지 발생되는 힘을 구하는 작업을 말한다.

$$\frac{\partial W_j}{\partial q_i} = \frac{\partial W_i}{\partial q_i} \cdot {}^i W_j$$

■ 사용하면

$$\begin{aligned} F_i &= \sum_{j=i}^n [\operatorname{tr}(\frac{\partial W_i}{\partial q_j} {}^i W_j J_i \ddot{W}_j^T) - m_i g^T \frac{\partial W_i}{\partial q_j} {}^i W_j \dot{r}_j] \\ &= \operatorname{tr}(\frac{\partial W_i}{\partial q_i} \sum_{j=i}^n {}^i W_j J_i \ddot{W}_j^T) - g^T \frac{\partial W_i}{\partial q_i} \sum_{j=i}^n m_j {}^i W_j \dot{r}_j \end{aligned} \quad (1)$$

윗 식을 다음과 같은 recursion을 사용하면

$$\begin{aligned} D_i &= \sum_{j=i}^n {}^i W_j J_i \ddot{W}_j^T \\ &= {}^i W_i J_i \ddot{W}_i^T + \sum_{j=i+1}^n A_{i+1} \cdot {}^{i+1} W_j J_j \ddot{W}_j^T \\ &= J_i \ddot{W}_i^T + A_{i+1} D_{i+1} \\ c_i &= \sum_{j=i}^n m_j {}^i W_j \dot{r}_j \\ &= m_i \dot{r}_i + A_{i+1} c_{i+1} \end{aligned}$$

D_i 와 c_i ■ (1)식에 대입하면 F_i 는 다음과 같다.

$$F_i = \operatorname{tr}(\frac{\partial W_i}{\partial q_i} D_i) - g^T \frac{\partial W_i}{\partial q_i} c_i$$

III-3. Recursive Lagrangian dynamics with 3X3 matrices

앞의 두 방식에서는 계수의 크기를 줄임으로써 계산량을 줄였다. 또 link matrix를 4X4 rotation-translation matrix 대신 3X3 matrix로 쓰면 계산량을 더 많이 줄일 수가 있다. 4X4 matrices는 dynamics를 set up 시키는 데는 매우 편리하지만 rotation과 translation이 함께 쓰여지고, rotation과 translation을 포함하지 않는 4X1 행렬 때문에 계산량이 추가가 된다. ${}^i A_j$ ■ $j-1$ 과 j 좌표

계의 orientation을 나타내는 3X3 matrix라 하자. 이때 j 축 좌표계의 orientation을 ν 라 하면 $i^{-1}\nu = A_j i\nu$ 이다.

Fig-1과 같은 벡터들을 다음과 같이 정의 한다.

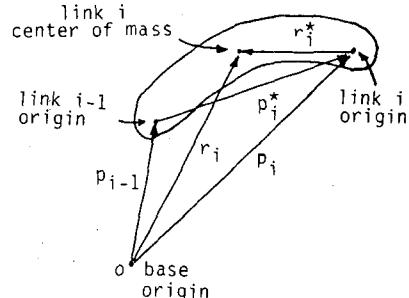


Fig-1. 좌표에서의 parameter

p_i : base 좌표계의 원점에서 joint i 좌표계의 원점까지의 vector

p_i^* : $i-1$ 좌표계의 원점에서 joint i 좌표계의 원점까지의 vector

r_i : base 좌표계의 원점에서 link i 의 질량 중심까지의 vector

r_i^* : i 좌표계의 원점에서 link i 의 질량 중심까지의 vector

$n_i = r_i^*/m_i$

link i 상의 모든 임자의 운동 에너지는 다음과 같이 나타낼 수 있다.

$$k_i = 1/2 \operatorname{tr}(\dot{r}_i \dot{r}_i^T) dm$$

$$= 1/2 \operatorname{tr}(\dot{p}_i \dot{p}_i^T + 2\dot{W}_i {}^i r_i^* \dot{p}_i^T + \dot{W}_i {}^i r_i^* W_i) dm$$

여기서 $r_i = p_i + W_i {}^i r_i^*$, $\dot{r}_i = \dot{p}_i + \dot{W}_i {}^i r_i^*$ 이다.

link i 상의 모든 임자의 운동 에너지를 구하기 위해 적분하면 K_i 는

$$K_i = 1/2 \operatorname{tr}(m_i \dot{p}_i \dot{p}_i^T + 2\dot{W}_i {}^i n_i \dot{p}_i^T + \dot{W}_i {}^i r_i^* \dot{W}_i^T)$$

이다. 여기서 m_i 는 link i 의 질량, ${}^i n_i = \int {}^i r_i dm$, ${}^i n_i/m_i$ 는 질량 중심의 벡터, $J_i = \int {}^i r_i^* {}^i r_i^* dm$ 이다. link의 운동 에너지는

$$K = \sum_{j=1}^n K_j = 1/2 \sum_{j=1}^n \operatorname{tr}(m_j \dot{p}_j \dot{p}_j^T + 2\dot{W}_j {}^i n_j \dot{p}_j^T + \dot{W}_j {}^i r_j^* \dot{W}_j^T)$$

이고, 또 link의 위치 에너지는

$$P = - \sum_{j=1}^n m_j g^T W_j {}^i r_j$$

이다. $L = K - P$ 이므로 힘 F_i 는

$$F_i = \frac{d}{dt} (\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i}$$

$$= \frac{d}{dt} (\frac{\partial K}{\partial \dot{q}_i}) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i}$$

이다. K 와 P 값을 대입하여 풀면

$$\begin{aligned} F_i &= \sum_{j=i}^n [\operatorname{tr}(\frac{\partial P_j}{\partial q_i} \dot{P}_j^T + \frac{\partial P_j}{\partial q_i} {}^i n_j \dot{W}_j^T + \frac{\partial W_j}{\partial q_i} {}^i n_j \dot{P}_j^T + \frac{\partial W_j}{\partial q_i} J_j \ddot{W}_j^T) - m_j g^T \frac{\partial W_j}{\partial q_i} {}^i r_j] \end{aligned}$$

가 된다. 여기서, W_j 는 3X3 matrix이다.

또

$$\ddot{P}_j = \ddot{P}_{j-1} - \ddot{W}_j {}^i p_j^*$$

다시 D_i 와 C_i 를 사용하면 F_i 는

$$F_i = \text{tr} \left(\frac{\partial W_i}{\partial q_i} D_i \right) - g^T \frac{\partial W_i}{\partial q_i} c_i$$

■ 알는다.

IV. Newton-Euler equations

N-E equations는 식들이 recursive한 관계를 갖고 있기 때문에 계산량은 상당히 줄어드는 장점이 있지만, 운동 방정식의 각 항을 간의 상호관계가 모호하므로 robot의 제어 시스템을 설계하는데는 많은 어려움이 따른다.

IV-1. Kinematics of the Links

link i 와 i+1의 좌표계를 coordinate (x_i, y_i, z_i) 와 $(x_{i+1}, y_{i+1}, z_{i+1})$ 라 하자.
이때의 선형 속도, 선형 가속도, 각속도, 각각속도는

$$\dot{v}_i = \begin{bmatrix} \omega_i \times p_i^* + v_{i-1} \\ z_{i-1}\dot{q}_i + \omega_i \times p_i^* + v_{i-1} \end{bmatrix} ; \text{ rotation 시} \quad \dots \dots \dots (2)$$

$$\ddot{v}_i = \begin{bmatrix} \dot{\omega}_i \times p_i^* + \omega_i \times (\omega_i \times p_i^*) + \ddot{v}_{i-1} \\ z_{i-1}\ddot{q}_i + \dot{\omega}_i \times p_i^* + 2\omega_i \times (z_{i-1}\dot{q}_i) + \omega_i \times (\omega_i \times p_i^*) + \ddot{v}_{i-1} \end{bmatrix} ; \text{ translation 시} \quad \dots \dots \dots (3)$$

$$\omega_{i+1} = \begin{bmatrix} \omega_i + z_i \dot{q}_{i+1} \\ \omega_i \end{bmatrix} ; \text{ rotation 시}$$

$$\dot{\omega}_{i+1} = \begin{bmatrix} \dot{\omega}_i + z_i \ddot{q}_{i+1} + \omega_i \times (z_i \dot{q}_{i+1}) \\ \dot{\omega}_i \end{bmatrix} ; \text{ rotation 시}$$

과 같다.

VI-2. Robot 팔의 운동 방정식

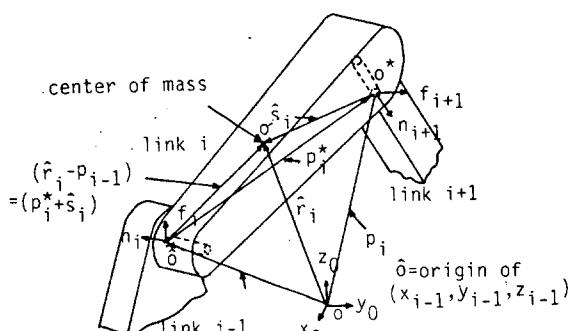


Fig-2. Link에서의 parameter

Fig-2와 같은 robot의 각 link에 D'Alembert's Principle를 사용함으로써 robot 팔의 운동을 나타낼 수 있다.

다음 기호를 아래와 같이 정의한다.

m_i = total mass of link i

\bar{r}_i = position of the center of mass of link i
of mass of link with reference to base
coordinates

\bar{v}_i = $d\bar{r}_i/dt$, linear velocity of the center
with reference to base coordinates

F_i = total external vector force exerted
on link i

J_i = inertia matrix of link i about its
center of mass in (x_0, y_0, z_0)

모든 joint의 viscous damping effect ■ 무시
하면 F_i, N_i 는

$$F_i = d(m_i \bar{v}_i)/dt = m_i \ddot{v}_i$$

$$N_i = d(J_i \omega_i)/dt = J_i \ddot{\omega}_i + \omega_i \times (J_i \omega_i)$$

로 나타내어진다.

식 (2)와 (3)을 이용하면, link의 질량중심의 선형속도와 가속도를 얻는다.

$$\bar{v}_i = \omega_i \times s_i + v_i$$

$$\ddot{v}_i = \omega_i \times s_i + \omega_i \times (\omega_i \times s_i) + \ddot{v}_i$$

이웃한 link들인 link i-1과 link i+1에 의해
여인가된 총 외부 힘 F_i 와 모우먼트 N_i 는

$$F_i = f_i - f_{i+1} \quad \dots \dots \dots (4)$$

$$N_i = n_{i-1}n_{i+1} + (p_{i-1} - \bar{r}_i) \times F_i - p_i^* \times f_{i+1} \quad (5)$$

이다. 이때 $(\bar{r}_i - p_{i-1}) = (p_i^* + s_i)$ 의 관계를
이용하면 식(4)와 (5)는

$$f_i = F_i + f_{i+1} = m_i \bar{v}_i + f_{i+1}$$

$$n_i = n_{i+1} + p_i \times f_{i+1} + (p_i^* + s_i) \times F_i + N_i$$

로 된다. 만일 joint i가 rotation이라면,
이때 link i는 좌표계 $(x_{i-1}, y_{i-1}, z_{i-1})$ 에서
 z_{i-1} 축에 대하여 q_i radian 만큼 회전된다. 그러
므로 joint i의 입력 torque는 z_{i-1} 축의 ni 와
좌표계에서의 viscous damping moment의 합이
된다. 하지만 joint i가 translation이라면
좌표계 $(x_{i+1}, y_{i+1}, z_{i+1})$ 의 z_{i-1} 축을 따라 q_i
만큼 translation 된다. 그러므로 joint i의
입력 torque/moment는

$$\tau_i = \begin{cases} n_i^T z_{i-1} + b_i \dot{q}_i & ; \text{link i가 rotation 시} \\ f_i^T z_{i-1} + b_i \dot{q}_i & ; \text{link i가 translation 시} \end{cases}$$

여기서 b_i 는 joint i에서의 viscous damping
coefficient이다. 만일 base가 고정되어 있다면
 $\omega_0 = \dot{\omega}_0 = 0$ 이고 $v_0 = 0$, 또 \dot{v}_0 는

$$\dot{v}_0 = g = \begin{bmatrix} gx \\ gy \\ gz \end{bmatrix}$$

이다. 여기서 $|g| = 9.8062 \text{ m/sec}^2$

V. 개선된 Newton-Euler equations.

IV장에서 N-E equations를 이용한 dynamic
운동 방정식을 다루었다. 그러나 이 방법은
관성 행렬 J_i 와 physical geometric parameter
들인 $\bar{r}_i, s_i, p_{i+1}, p_i^*$ 등이 base 좌표를 기준 좌
표로 하고 있기 때문에, robot 팔이 움직일때
이들 값이 변한다는 점이다.

이런 걸점을 보완하기 위하여 위치와 속도,
가속도, 관성 행렬, 또 각 link의 질량중심
을 각 link 좌표계를 기준으로 하여 구하면,
관식의 성질과 joint torque를 재계적으로 계
산하는 방법들을 마련해 놓았을 때 계산이 훨씬 간
단해지는 점이다.

이 방식의 또 다른 장점은 torque를 계산하
는 시간이 robot 팔의 joint의 수에 따라 선형
적으로 비례하고, 어느 로봇 팔의 구조에 상
관없이 쓸 수 있다는 점이다. R_i 를 좌표계
 $(x_{i-1}, y_{i-1}, z_{i-1})$ 에 대한 좌표계 (x_i, y_i, z_i) 의
변환을 나타내는 3×3 rotation matrix라 하자.

이것은 다음과 같은 관계를 가지고 있다.

여기서

$${}^{i+1}R_i = \begin{bmatrix} \cos \theta_i & -\cos \alpha_i \sin \theta_i & \sin \alpha_i \sin \theta_i \\ \sin \theta_i & \cos \alpha_i \cos \theta_i & -\sin \alpha_i \cos \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i \end{bmatrix}$$

이다. 이제 $\omega_{0i+1}, \dot{\omega}_{0i+1}, \dot{v}_i, \ddot{v}_i, p_i^*, s_i, F_i, N_i, f_i, n_i, \tau_i$ 를 계산하는 대신에 ${}^iR_0\omega_{i+1}, {}^iR_0\dot{\omega}_{i+1}, {}^iR_0\ddot{\omega}_{i+1}, {}^iR_0\dot{v}_{i+1}, {}^iR_0\ddot{v}_{i+1}, {}^iR_0F_i, {}^iR_0N_i, {}^iR_0f_i, {}^iR_0n_i, {}^iR_0\tau_i$ 를 계산한다.

$${}^{i+1}R_0\omega_{i+1} = \begin{bmatrix} {}^{i+1}R_i ({}^iR_0\omega_i + z_0\dot{q}_{i+1}) & ; \text{rotation} \\ {}^{i+1}R_i ({}^iR_0\dot{v}_i) & ; \text{translation} \end{bmatrix}$$

$${}^{i+1}R_0\dot{\omega}_{i+1} =$$

$$\begin{bmatrix} {}^{i+1}R_i [{}^iR_0\dot{\omega}_i + z_0\ddot{q}_{i+1} + {}^iR_0\omega_i \times (z_0\dot{q}_{i+1})] & ; \text{rotation} \\ {}^{i+1}R_i ({}^iR_0\dot{v}_i) & ; \text{translation} \end{bmatrix}$$

$${}^{i+1}R_0v_{i+1} =$$

$$\begin{bmatrix} ({}^{i+1}R_0\omega_{i+1}) \times ({}^{i+1}R_0p_{i+1}^*) + {}^{i+1}R_i({}^iR_0v_i) & ; \text{rotation} \\ {}^{i+1}R_0(z_0\dot{q}_{i+1} + {}^iR_0v_i) + ({}^{i+1}R_0\omega_{i+1}) \times ({}^{i+1}R_0p_{i+1}^*) & ; \text{translation} \end{bmatrix}$$

$${}^{i+1}R_0\dot{v}_{i+1} =$$

$$\begin{bmatrix} ({}^{i+1}R_0\dot{\omega}_{i+1}) \times ({}^{i+1}R_0p_{i+1}^*) + ({}^{i+1}R_0\omega_{i+1}) \times \\ [({}^{i+1}R_0\omega_{i+1}) \times ({}^{i+1}R_0p_{i+1}^*)] + {}^{i+1}R_i({}^iR_0\dot{v}_i) & ; \text{rotation} \\ {}^{i+1}R_0(z_0\ddot{q}_{i+1} + {}^iR_0\dot{v}_i) + {}^{i+1}R_0\dot{\omega}_{i+1} \times ({}^{i+1}R_0p_{i+1}^*) \\ + 2({{}^{i+1}R_0\omega_{i+1}}) \times ({}^{i+1}R_0z_0\dot{q}_{i+1}) + ({}^{i+1}R_0\omega_{i+1}) \\ \times [({}^{i+1}R_0\omega_{i+1}) \times ({}^{i+1}R_0p_{i+1}^*)] & ; \text{translation} \end{bmatrix}$$

$${}^iR_0F_i = m_i {}^iR_0\dot{v}_i$$

$${}^iR_0N_i = ({}^iR_0J_i{}^0R_i)({}^iR_0\dot{\omega}_i) + ({}^iR_0\dot{\omega}_i) \times \\ [({}^iR_0J_i{}^0R_i)({}^iR_0\omega_i)]$$

$${}^iR_0\ddot{v}_i = ({}^iR_0\omega_i) \times ({}^iR_0\ddot{s}_i) + {}^iR_0v_i$$

$${}^iR_0\dot{\ddot{v}}_i = ({}^iR_0\dot{\omega}_i) \times ({}^iR_0\ddot{s}_i) + ({}^iR_0\omega_i) \times \\ [({}^iR_0\omega_i) \times ({}^iR_0\ddot{s}_i)] + {}^iR_0\ddot{v}_i$$

$${}^iR_0f_i = {}^iR_{i+1}({}^{i+1}R_0f_{i+1}) + {}^iR_0F_i$$

$${}^iR_0n_i = {}^iR_{i+1}[{}^{i+1}R_0n_{i+1} + ({}^{i+1}R_0p_i^* \times \\ ({}^{i+1}R_0f_{i+1})) + ({}^{i+1}R_0p_i^* + {}^iR_0\ddot{s}_i) \times \\ ({}^iR_0F_i) + {}^iR_0N_i]$$

$$\tau_i = \begin{bmatrix} ({}^iR_0n_i)^T ({}^iR_{i+1}z_0) + b_i q_i & ; \text{rotation} \\ ({}^iR_0f_i)^T ({}^iR_{i+1}z_0) + b_i q_i & ; \text{translation} \end{bmatrix}$$

여기서 $z_0=(0, 0, 1)$ 이다. ${}^iR_0\ddot{s}_i$ 는 좌표계(x_i, y_i, z_i)에 대한 link i의 질량중심이고, ${}^iR_0p_i^*$ 는 좌표계원점($x_{i+1}, y_{i+1}, z_{i+1}$)에서 (x_i, y_i, z_i)까지의 위치이고 이것은 다음과 같다.

$${}^iR_0p_i^* = \begin{bmatrix} \alpha_i \\ r_i \sin \alpha_i \\ r_i \cos \alpha_i \end{bmatrix}$$

VI. 계산 Algorithm

본 논문에서 사용한 계산 Algorithm은 다음과 같다.

VI-1. L-E methods 계산 Algorithm

- 1) 질량 m_i , 관성 J_i , i_{ri} 값을 입력한다.
- 2) link parameter 값을 입력한다.
- 3) joint i의 위치 q_i , 속도 \dot{q}_i , 가속도 \ddot{q}_i 값을 입력한다.
- 4) link $i-1$ 과 link i 의 관계를 나타내는 4×4 homogeneous transformation matrix ${}^{i-1}A_i$ 를 계산한다.
- 5) joint j의 변화가 link i에 미치는 변화 U_{ij} 를 계산한다.
- 6) robot 팔의 총 운동 에너지 K를 계산한다.
- 7) robot 팔의 총 위치 에너지 P를 계산한다.
- 8) nx1 풍력 벡터 G_i 를 계산한다.
- 9) 힘 F_i 를 계산한다.

VI-2. Recursive L-E methods 계산 Algorithm

- 1) 질량 m_i , 관성 J_i , i_{ri} 값을 입력한다.
- 2) link parameter 를 입력한다.
- 3) joint i의 위치 q_i , 속도 \dot{q}_i , 가속도 \ddot{q}_i 값을 입력한다.
- 4) rotation matrix A_i 를 계산한다.
- 5) $W_i, \dot{W}_i, \ddot{W}_i$ 의 값을 계산한다.
- 6) $p_i, \dot{p}_i, \ddot{p}_i, p_i^*$ 값을 계산한다.
- 7) D_i, c_i 를 계산한다.
- 8) 힘 F_i 를 계산한다.

VI-3. N-E methods 계산 Algorithm

- 1) 초기 값을 입력한다.
 $\omega_0 = \dot{\omega}_0 = 0, v_0 = 0,$
 $\ddot{v}_0 = (0, 0, 9.80621 \text{ m/sec}^2)^T$
- 2) link parameter 값을 입력한다.
- 3) joint i의 위치 q_i , 속도 \dot{q}_i , 가속도 \ddot{q}_i 를 입력한다.
- 4) $i=1$ 로 놓는다.
- 5) link의 각속도 ω_i , 각가속도 $\dot{\omega}_i$, 선형 속도 v_i , 선형 가속도 \ddot{v}_i 를 계산한다.
- 6) base좌표계에 대한 link i의 질량 중심의 선형 속도 \ddot{v}_i 를 계산한다.
- 7) link i의 외부 힘 F_i 와 외부 모멘트 N_i 를 계산한다.
- 8) 만일 $i=n$ 이면 stop 하고, $i=n$ 이 아니면 $i=i+1$ 로 놓고 4)로 돌아간다.
- 9) link $i-1$ 에 의하여 link i에 인가된 힘 f_i 와 모멘트 n_i 를 구한다.
- 10) 만일 $i=1$ 이면 stop 하고, $i=1$ 이 아니면 $i=i-1$ 로 놓고 8)로 돌아간다.

VI-4. improved N-E methods 계산 Algorithm

- 1) 초기 값을 입력한다.
 $\omega_0 = \dot{\omega}_0 = 0, v_0 = 0,$
 $\ddot{v}_0 = (0, 0, 9.80621 \text{ m/sec}^2)^T$
- 2) link parameter 값을 입력한다.
- 3) joint i의 위치 q_i , 속도 \dot{q}_i , 가속도 \ddot{q}_i 값을 입력한다.
- 4) $i=1$ 로 놓는다.
- 5) ${}^iR_0\omega_i, {}^iR_0\dot{\omega}_i, {}^iR_0v_i, {}^iR_0\ddot{v}_i$ 를 계산한다.
- 6) joint i에 대한 link i의 질량 중심의 선형 속도 ${}^iR_0\ddot{v}_i$ 를 계산한다.
- 7) iR_0F_i 와 iR_0N_i 를 계산한다.
- 8) 만일 $i=n$ 이면 stop 하고, $i=n$ 이 아니면 $i=i+1$ 로 놓고 5)로 돌아간다.
- 9) iR_0f_i 와 iR_0n_i 를 계산한다.
- 10) 만일 $i=1$ 이면 stop 하고, $i=1$ 이 아니면 $i=i-1$ 로 놓고 9)로 돌아간다.

VII. 실험 및 고찰

본 논문에서는 L-E equations, Recursive L-E equations, N-E equations의 유도와 improved N-E computational scheme에 대하여 알아보았고, 이를 program화하여 각 joint의 토오크를 알기까지의 실행시간을 살펴보았다.

본 연구에서 사용한 computer는 IBM-PC/XT를 이용하였으며 개발한 program은 PASCAL로 작성하였다. Fig-3은 본 연구에 사용한 PUMA-600 Robot의 외관을 보여주며 이에 대한 제원을 Table-1에 서제시하였다.

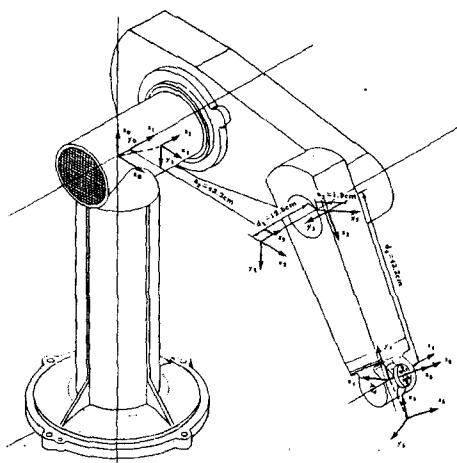


Fig-3. PUMA-600

Table-1. Link Parameter for PUMA-600

Joint	q^o	θ^o	d	a
1	-90	θ_1	0	0
2	0	θ_2	0	a_2
3	90	θ_3	d_1	a_3
4	-90	θ_4	d_1	0
5	90	θ_5	0	0
6	0	θ_6	0	0

$$a_2 = 17.00(\text{in}) = 43.2(\text{cm}), a_3 = 0.75(\text{in}) = 1.9(\text{cm}), d_3 = 4.937(\text{in}) = 12.5(\text{cm}), d_4 = 17.00(\text{in}) = 43.2(\text{cm})$$

내가지 운동 방정식에 대한 program을 실행시키고 결과로 Table-2와 같은 실행 시간을 알아내었다.

Table-2. 내가지 방식의 실행시간

method	execution time(sec)
L-E equations	21
Recursive L-E equations	8.5
N-E equations	1.6
Improved N-E equations	1

VIII. 결론

본 논문에서는 L-E equations, Recursive L-E equations, N-E equations와 improved N-E computational scheme에 대하여 알아보았고, 이를 program화하여 각 joint의 토오크를 구하기까지의 실행시간에 대하여 비교하였다. robot의 dynamics를 고려한 내가지 방식을 program화하여 joint의 토오크를 구하기까지의 실행시간을 비교한 바 L-E equations를 사용한 방식들보다 N-E equations를 사용한 방식이 훨씬 효율적임을 알 수 있었다. 또 N-E 방식에서도 Improved N-E 방식이 실행시간이 짧음을 알아내었다.

본 논문에서는 robot의 Dynamics를 고려한 방정식들에 대하여 연구하였으나 기어의 전기 특성이나 마찰력들은 고려하지 않았으므로 이 효과들을 고려한 식의 유도에 관한 연구가 되어야 하겠고, robot 팔의 파라미터들에 관한 정확한 값을 얻어내는데도 각별한 관심을 필요로 한다. 또 로보트의 특성을 고려한 controller의 제작과 robot를 위한 sensor system에 대하여도 많은 연구가 있어야겠다.

참고 문헌

- [1] John. J. Craig. "Introduction to Robotics". ADDISON-WESLEY. 1986.
- [2] C.S.G. Lee. "Robot Arm Dynamics". IEEE. 1983.
- [3] John. M. Hollerbach. "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity". IEEE Transaction on System, Man, and Cybernetics, Vol. SMC-10. No.11. November. 1980.
- [4] J.Y.S. Luh. "On-Line Computational Scheme for Mechanical Manipulators". Transaction of ASME. 1980.
- [5] M.W. Walker. "Efficient Dynamic Simulation Of Robotic Mechanisms". Journal of Dynamic Systems, Measurement, and Control. Vol. 104. 1982.
- [6] Richard. P. Paul. "Robot Manipulators". MIT Press. 1981.