論文 88-25-4-4

여유 자유도를 갖는 머니퓰레이터의 기구학적 제어를 위한 효율적 계산 방법

(An Efficient Computation Method for Kinematic Control of Redundant Manipulators)

李 京 秀*, 徐 一 弘**, 林 俊 弘***

(Kyoung Soo Lee, Il Hong Suh and Joon hong Lim)

要 約

본 논문에서는 여유 자유도를 갖는 머니퓰레이터를 기구학적으로 제어하는 방법에 관하여 논하고자 한다. 직각 좌표계에서 머니퓰레이터의 주어진 경로를 추적하기 위한 관절 변수의 해를 구하는 효율적인 계산 방법을 제안한다. 제안 방법에서는 경로 오차량이 규정된 허용 오차를 초과할 경우에만 자코비안 행렬과 그의 가상 역행렬을 간헐적으로 계산하게 된다. 따라서 계산상의 부담이 상당히 줄어들고 동시에 오차량도 규정된 범위내에 유지된다.

본 방법의 타당성과 효율성을 시뮬레이션 결과에서 보였으며 여유 자유도는 특이 상태를 회피하도록 이용하였다.

Abstract

A kinematic control for redundant manipulators is considered. An efficient computation method is proposed to determine the joint variable solutions for a given Cartesian path of the end effector.

In the proposed method, the Jacobian matrix and its pseudoinverse matrix are calculated intermittently only when the errors exceed the prescribed tolerance. Thereby, the computational burdens are greatly reduced, and at the same time, the errors are maintained within a tolerable range.

To show the effectiveness of the method, the result of the simulation is provided in which the redundancy of the manipulator is resolved to avoid the singularity.

*正會員, 漢陽大學校 電子通信工學科
(Dept. of Elec. Commu., Hanyang Univ.)
**正會員, 漢陽大學校 電子工學科
(Dept. of Elec. Eng., Hanyang Univ.)
***正會員, 韓國航空大學 航空電子工學科
(Dept. of Avionics, Hankook Aviation College)
接受日字 : 1987年 6月 22日

Ⅰ. 서 론

공장 자동화의 일환으로 여러가지 제조분야에서 산업용 로보트가 사용되는 경우가 많아짐에 따라 최근에는 산업용 로보트의 그 기능의 다양성과 유동성에 대한 요구가 증가하고 있다. 그러나 산업계에서 사용되는 대부분의 로보트는 공간상에서 위치와 자세를 모두 제어할 경우에도 6관절을 넘지 않는 필요한

최소의 자유도를 이용하기 때문에 그기능에 대하여 더욱 많은 역할을 기대하기가 어렵다. 따라서 이러한 요구를 충족시키기 위하여 로보트 머니퓰레이터에 더욱 많은 자유도를 주고 있는데, 이와 같이 어떠한 과제 수행에 필요한 자유도 이상의 자유도를 갖는 로보트를 그 과제에 대하여 여유 자유도를 가지고 있다고 말한다.

여유 자유도를 갖는 머니퓰레이터를 제어하는 방법에 관한 연구들로는 동작 가능도 지수 (Manipulability) 를 측정하여 그 크기를 크게하여 줌으로써 특이 상태(singular state)를 회피하는 연구[1] 와 장애물을 피하기 위한 원하는 자세를 주고 머니퓰레이터로 하여금 그 자세를 취하게하여 충돌을 회피하는 방법[1,2] 물체로부터의 거리를 최대로 하여서 장애물을 피하기 위한 관절 각속도를 결정하는 방법[3] 등 부차적인 과제를 수행하기 위한 연구들이 있으며, 역 기구학적인 해를 일반적인 머니퓰레이터에 대하여 수치적인 방법으로 구하려는 연구도 진행되었다.[4]

로보트 머니퓰레이터의 경로 제어시에는 머니퓰레이터 손의 end effector 의 원하는 경로에 대응 하는 각 관절의 궤적을 구하여야 한다. 즉, 머니퓰레이터 손의 위치와 자세를 알고서 거기에 대응하는 각 관절의 각도 값을 매 순간 구하여야 한다. 그러나 일반적인 머니퓰레이터의 기구학적 방정식은 비선형이고, 여유 자유도가 있을 경우에는 역 기구학적 해도 무수히 많게 된다. 따라서 머니퓰레이터 손의 경로를 작은 구간으로 나누고 기구학적 방정식을 jacobian matrix를 이용하여 선형 관계식으로 근사화 시킴으로써 각 관절의 각도 값을 구한다. 무수히 많은 해 중에서 부차적인 과제를 수행하는 성능 지수가 최소 혹은 최대가 되는 해를 구하게 된다.[5]

이러한 경로 제어 방법은 자코비안 행렬 (jacobian matrix) 과 그 가상 역 행렬(pseudoinverse matrix) 을 매 구간마다 구하여야 하고, 또 근사 식의 오차를 줄이기 위하여는 가능한 많은 구간으로 경로를 나누어야 하므로 많은 양의 계산을 요구하게 된다. 계산량을 줄이기 위하여 매 구간 마다 계산하지 않고 어떠한 일정한 간격마다 계산하는 방법이 제안되었다.[6] 그러나, 계산량은 줄게되나 오차가 누적되어 허용 오차를 넘어 서는 경우가 있게된다. 따라서 본 논문에서는 이러한 단점들을 보완하기 위하여 보다 효율적이고 오차를 줄이고 풀기 위한 방법을 제안한다. 제안 방법에서는 자코비안과 가상 역 행렬들을 매 구간 구하지 않고 경로의 오차가 누적되는 양을 계산하여 그것이 허용 오차를 넘는 구간에서만 간헐적으로 계산한다. 그러므로 계산량도 줄이고 또한 오차가 언제나 허용된 범위 내에 있게 되는 효율적인 계산 방법이다. 또한 자코비안과 가상 역 행렬도 효율적인 계산을 위하여 순환 관계식(recursion relation) 을 이용하였다.

제안된 방법의 타당성과 효율성을 보이기 위하여 응용 예를 시뮬레이션하였다. 여유 자유도를 이용하여서는 로보트 동작 중에 발생하는 특이 상태 (singularity) 를 회피하는 부차적인 과제를 수행하였다.

Ⅱ. 여유 자유도를 갖는 로보트의 기구학적 방정식 및 그 일반해

일반적으로 머니퓰레이터의 mechanism 을 나타낼 때에는 수행하고자 하는 과제를 나타내는 벡터와 관절의 벡터로 표시한다. n관절 로보트 즉, n개의 자유도(degree of freedom)를 갖는 로보트를 생각할 때, 관절각 벡터(joint vector)는 $\theta=(\theta_1, \theta_2, \cdots, \theta_n)$로 표시하고 원하는 작업과제(task)는 m 차원의 위치 및 자세의 벡터(manipulation vector) $r=(r_1, r_2, \cdots, r_n)$로 표시 한다. 그러면 r과 θ의 관계식은

$$r=f(\theta) \qquad (1)$$

로 표시할 수 있다. 여기서 $[\cdot]^T$는 행렬 $[\cdot]$의 transpose를 의미한다. 식 (1)을 시간에 관하여 양변을 각각 미분하면 속도 관계식

$$\dot{r}=J(\theta)\dot{\theta} \qquad (2)$$

이 된다. 식 (2)에서 $J(\theta)$는 $\partial f/\partial\theta$로서 m×n 차원 자코비안 행렬(jacobian matrix)이라고 하며, 비선형 관계식인 식 (1)을 선형화 시키는 계수가 된다. 여기서 문제는 주어진 $\dot{r}$에 대하여 $\dot{\theta}$를 찾는 것으로서, 그 해는 J의 차원(dimension) 과 rank 에 따라 달라진다. n>m 일 경우, 머니퓰레이터는 과제에 대하여 여유 자유도를 갖게 되고 식 (2)의 해는 무수히 많이 존재하게 된다. 따라서 원하는 부차 적인 과제를 수행할 수 있는 해를 선택한다. 식 (2)의 해는 일반적으로

$$\dot{\theta}=J^+(\theta)\ \dot{r}+(I-J^+(\theta)\ J(\theta))\ Z(\theta) \qquad (3)$$

로 주어진다. 여기서 J^+는 자코비안 행렬 J의 가상 역 행렬(pseudo-matrix)[7] 이라고 하며 J가 완전 랭크(full-rank) 이면

$$J^+=J^T(JJ^T)^{-1} \qquad (4)$$

로 주어진다. 또한, $Z(\theta)$는 임의의 n차원 열 벡터 (column vector) 이다. 이 항을 이용하여 우리가 원

하는 부차적인 과제 (secondary task) 를 수행할 수 있다.

머니퓰레이터의 경로 제어시, 전 경로를 n개의 구간으로 나누고 k 번째 구간 (k=1,2,…, N) 에서 식 (3) 으로써 θ를 구하려면 이산화 (discretization) 시킨 형태의 식

$$\theta(k)=\theta(k-1)+\Delta\theta(k-1) \qquad (5)$$

을 사용하여야 한다. 따라서 k 값이 변할 때마다 그것에 대응하는 J 및 J^+를 구하여야만 $\Delta\theta$를 구하므로 θ값을 구할 수 있다. 그러므로 J 및 J^+를 일반적이고도 계산의 효율성이 좋게 구하는 알고리즘의 개발이 중대한 영향을 미치게 된다. 이제 위에서 기술한 바와 같이, J 및 J^+를 컴퓨터를 이용하여 순환적으로 구하는 방법 및 식 (2)를 가장 효율적으로 풀기 위한 방법들을 조사하고 이를 적용한 컴퓨터 시뮬레이션을 통하여 그의 효율성을 검토하고자 한다.

Ⅲ. 자코비안 행렬 및 가상 역 변환의 순환적 계산 방법

자코비안 행렬은 다차원 형태의 도함수이다. 따라서 머니퓰레이터의 기구학적 방정식이 주어지면 편미분을 이용하여 자코비안 행렬을 closed form 으로 구할 수 있다.[8,9] 그러나 자코비안 행렬을 이와 같이 closed form으로 구하는 방법은 많은 관절을 가지는 redundant manipulator에서는 매우 지루하고, 오류를 범하기 쉬운 과정을 필요로 한다. 또한 많은 연산 과정을 필요로 하므로 계산상의 효율성도 떨어지게 된다. 따라서 반복적이고 효율적인 계산에 의해 자코비안 행렬을 구하는 방법에 대하여 많은 연구가 이루어 졌다.[10,11] 본 논문에서는 [10] 에서 제시한 방법을 사용하여 자코비안 행렬을 구한다. 또한 가상 역 행렬도[7] 에서 제시한 바와 같이 순환형태로 구하여 계산의 효율성을 높이고자 한다.

1. 순환적 자코비안 행렬 계산 방법

그림 1은 i와 i+1 번째의 (i=1,2,…, n) 두 링크 간의 관계를 나타낸다. 여기서 αi는 링크의 길이 이고, si는 두 좌표계 간의 차감거리를 의미한다. 또 αi는 좌표계 간의 비틀림 각도이고, θi는 관절의 회전각이다. 이러한 링크 매개 변수들 (link parameters) 로부터 두 좌표계 간의 관계는 로드리게스 방정식 (rodrigues' equation)[10] 을 이용하면 다음과 같이 나타내어 진다.

$$Zi=Zi+1\ \cos\alpha i+Yi+1\ \sin\alpha i$$

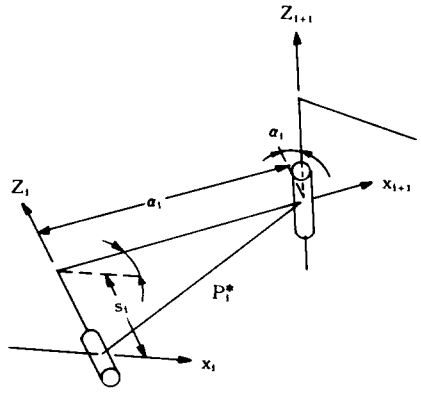

그림 1. 링크 매개 변수들의 정의.
Fig. 1. Definition of link parameters.

$$Xi=Xi+1\ \cos\theta i-Yi+1\ \sin\theta i\ \cos\alpha i+Zi+1\ \sin\theta i\ \sin\alpha i \qquad (6)$$
$$Yi=Xi+1\ \sin\theta i+Yi+1\ \cos\theta i\ \cos\alpha i-Zi+1\ \cos\theta i\ \sin\alpha i$$

또한 $\overset{*}{P}i$를 i 번째 좌표계 원점에서 i+1 번째 좌표계 원점까지의 거리 벡터라 하고 ${}^{n+1}Pi$ 를 i 번째 좌표계 원점에서 n+1 번째 좌표계 원점 (즉, end effector 의 좌표계) 까지의 거리 벡터라 하면, $\overset{*}{P}i$, ${}^{n+1}Pi$ 는

$$\overset{*}{P}i=Xi\ i+Yi\ Si\ \sin\alpha+Zi\ Si\ \cos\alpha i$$
$${}^{n+1}Pi={}^{n+1}Pi+1\ \overset{*}{P}i,\quad i=n,n-1,\cdots,1 \qquad (7)$$

로 주어진다.

(6)식과 (7)식으로 vector cross product를 이용하여 i 번째 관절에 의한 end effector 자코비안 행렬의 i 번째 열 (column) 을 구하면 다음과 같다.[10,12]

회전 관절일 경우 :

$$\{{}^{n+1}Ji\}=\begin{bmatrix}{}^{n+1}Zi\times{}^{n+1}Pi\\ {}^{n+1}Zi\end{bmatrix} \qquad (8)$$

미끄럼 관절일 경우 :

$$\{{}^{n+1}Ji\}=\begin{bmatrix}{}^{n+1}Zi\\ 0\end{bmatrix} \qquad (9)$$

여기서 ${}^{n+1}Z$ 는 n+1 좌표계에서 본 i번째 좌표계의 Z축 방향의 단위 vector 이다.

2. 가상 역 행렬 계산 방법

여유 자유도를 갖는 경우의 자코비안 역 행렬인, 가상 역 행렬도 (4)식과 같이 구하는 것은 많은 계산 시간과 양을 필요로 한다. 따라서 이것도 일반적인

경우에 적용하기 위하여 순환적인 방법을 사용하고자 한다. 이를 위하여 다음과 같은 알고리즘을 사용하였다.[7] 행렬 J의 k번째 열 vector를 j_k라 하고 J_k를 다음과 같이 정의한다.

$$J_k = [j_1 j_2 \cdots j_k] \tag{10}$$

그러면 J_k, $k=2,3,\cdots,n$의 가상 역 변환 행렬 J_k^+는

$$J_k^+ = \begin{bmatrix} J_{k-1}^+ - d_k c_k \\ b_k \end{bmatrix} \tag{11}$$

으로 표시된다. 여기서 d_k, c_k, b_k는 다음과 같이 나타낸다.

$$\begin{aligned} d_k &= J_{k-1}\, j_k \\ c_k &= j_k - J_{k-1}\, d_k \\ b_k &= c_k^+ + (1 - c_k^+ c_k)\ (1 + d_k^T d_k)^{-1}\ d_k^T J_{k-1}^+ \end{aligned} \tag{12}$$

따라서 $J^+ = J^+n$ 은 식 (10) ~ (12)를 이용하여 순환적으로 계산 한다.

Ⅳ. 경로 제어를 위한 계산 방법

본 장에서는 머니퓰레이터의 경로 제어시 식 (3)에 주어진 역 기구학적 방정식을 이용하여 θ를 수치적으로 구하는 방법을 오차 및 계산 속도의 관점에서 비교, 분석하고자 한다. 이를 위하여 먼저 기준 좌표계에서 머니퓰레이터의 end effector의 원하는 자세 및 위치를 주어진 경로의 각 구간에서 다음과 같이 표시한다.

$$r(1),\ r(2),\ r(3),\ \cdots r(N) \tag{13}$$

한편 머니퓰레이터가 구간 경로를 추적할 때, k번째 구간에서의 경로 오차 $e(k)$는 수치 해법에 의한 해 $s(k)$를 식 (1)에 대입하여 얻은 머니퓰레이터의 end effector의 위치 및 자세의 차이로 정의한다.

$$e(k) = r(k) - f(s(k)),\quad k=1,2,\cdots,N \tag{14}$$

따라서 k+1 번째 구간에서 머니퓰레이터 손의 end effector의 위치 및 자세가 $r(k+1)$이 되기 위한 관절각 $o(k+1)$은 다음과 같이 나타낼 수 있다.

$$\theta(k+1) = \theta(k) + J^+(\theta(k))\,\{r(k+1) - f(\theta(k))\} + \mathrm{red}(\theta(k)) \tag{15}$$

여기서 $\mathrm{Red}(\theta(k))$는 다음과 같이 주어진다.

$$\mathrm{Red}(\theta(k)) = (I - J^+(\theta(k))\ J(\theta(k)))\ \ Z(\theta(k)) \tag{16}$$

그러나 이와 같이 θ값을 구할 경우에는 머니퓰레이터가 구간 이동을 할 때 마다 J값을 매 구간 계속하여 구하여야 하므로 매우 많은 계산 시간과 양을 필요로 하게 된다. 따라서 본 논문에서는 매 구간 계산하지 않고 간헐적으로 J 및 J^+를 구하여 계산의 효율성을 높이고자 한다. 이 때 J의 계산 구간을 J. Furusho 와 S. Onishi[6] 는 일정한 간격으로 제안하고 있다. 그러나 일정한 간격마다 J 및 J^+를 간헐적으로 계산하게 되면 주어진 간격으로 진행하는 과정에서 어떤 구간에서는 정밀도가 낮아 계산 구간을 줄일 필요가 있는 경우나, 반대로 어떤 구간에서는 정밀도가 높아 계산의 효율성을 더 높일 수 있는 경우에도 이에 대응할 수 있는 조절이 불가하다. 따라서 본 논문에서는 계산 구간을 과거의 오차 크기에 따라 조절하는 방법을 제안한다. $r(k+1)$에 대응하는 $\theta(k+1)$ 식은 다음과 같이 표시한다.

$$\theta(k+1) = \theta(k) + H(k)\,\{r(k+1) - f(\theta(k))\} + \mathrm{red}(\theta(k)) \tag{17}$$

$$H(k) = \begin{bmatrix} J^+(\theta(1)) & \text{for } k=1,2,3,\cdots,m(2)-1 \\ J^+(\theta(m(2))) & \text{for } k=m(2),m(2)+1,\cdots,m(3)-1 \\ J^+(\theta(m(3))) & \text{for } k=m(3),m(3)+1,\cdots,m(4)-1 \\ \vdots & \\ \vdots & \\ \vdots & \end{bmatrix} \tag{18}$$

식 (18)에서, $m(i)$는 J와 J^+를 계산하는 구간시작을 의미하며 $m(i+1) - m(i)$는 $m(i)$ 번째에 구한 J와 J^+ 값을 전용하여 사용하는 구간수를 나타낸다. [6] 에서는 $m(i+1) - m(i)$가 상수의 정수이고 본 논문에서 변수로하여 다음과 같이 나타내고자 한다.

$$\begin{aligned} &m(i+1) - m(i) = V(i) \\ &V(i) = V(i-1) + \{\mathrm{Int}(\log\tfrac{\varepsilon}{\rho}) + 1\} \end{aligned} \tag{19}$$

ε : tolerance

$$\rho = \sqrt{\sum_{k=m(i+1)}^{m(i+1)} \frac{e(k)}{V(i)}}$$

$\mathrm{Int}(\cdot)$는 값의 정수 부분만을 취하게 하는 연산자(operator)이다. 그리고 ρ는 각 계산 구간에서의 경로의 root mean squared error 이다. 이 방법에서는 경로 오차가 허용되는 수치값이 주어지면 진행 과정에서 정하여지는 가변의 계산 구간 동안에 발생하는 각각의 구간 경로 오차량의 평균치인 ρ를 상대적으로 비교하여 계산 구간을 증감시키도록 하였다. 즉, ρ값이 ε보다 10^{-1}씩 작을 때마다 계산 구간을 하나씩 늘리도록 하였다. 같은 자리수의 값일 경우

에는 전 단계의 구간 수 보다 한 구간을 더 늘리기 위하여 1의 값을 더하였다.

이상의 머니퓰레이터를 제어하는 방법을 흐름도로 나타내면 그림 2와 같다.

V. 응용예 및 컴퓨터 시뮬레이션

1. 여유 자유도를 이용한 부차적 과제 수행(특이 상태 회피)

여유 자유도를 갖는 로보트 머니퓰레이터는 주어진 우선 과제(primal task)를 수행함은 물론 부차적인 과제(secondary task)도 수행할 수 있다. 즉, 식 (16)의 $Z(\theta)$를 부차적 과제를 수행할 수 있도록 선택할 수 있다. 부차적인 과제로는 [1]에서 제시한 특이 상태 회피를 고려한다. 이를 위하여 성능 지수 $P(\theta)$를 다음과 같이 선택한다.

$$P(\theta)=\sqrt{\det(J(\theta)J^{T}(\theta))} \qquad (20)$$

이를 동작가능도 지수(manipulability) 라고 한다.[1]

식 (20)의 $P(\theta)$를 최대로 하는 $Z(\theta)$를 구하면 특이 상태를 회피할 수 있고 이 $Z(\theta)$는 다음과 같다.[1,5]

$$Z(\theta)=K\frac{\partial P(\theta)}{\partial \theta} \quad , \quad (k>0) \qquad (21)$$

여기서 k는 임의의 상수이다.

2. 시뮬레이션 결과

4장에서의 방법들을 비교, 검토하기 위하여 그림 2에 보는 바와 같이 3개의 회전관절을 가지는 머니퓰레이터에 적용시켜 평면상에서 원하는 궤적을 추적하는 시뮬레이션을 수행하였다. 오차의 단위는 m이다. 제어하기 위해서는 2개의 자유도만으로 가능하기 때문에, 그림 2의 3자유도 로보트의 경우에는 1개의 자유도가 여유 자유도로 활용된다. 여유 자유도는 1에서 기술한 manipulability를 극대화하도록 이용하였다.

한편 경로는 (0.4460, 0.0915)와 (0.4460, −0.0085)을 잇는 직선으로 주었다. 따라서 총 10cm를 이동하는 것이며 계산 구간 수인 N은 100으로 선정하였다. 또한 제안 방법에서의 허용 오차는 5×10^{-4} 1×10^{-4} 5×10^{-5} 1×10^{-5}의 4경우에서 각각 시뮬레이션을수행하였다. 오차 ε의 단위는 m이다. 그리고 식 (21)의 k=5로 하였다. 그림 3은제안 방법으로 시뮬레이션 하였을 때 3자유도 로보트가 원하는 경로를 추적할 때의 자세를 보여주는 그림의 하나이다. 4의 경우가 서로 비슷한 결과의 그림을 보여주었다. Table 1에서는 6가지 경우에 대한 평균 경로 오차량과 계산되

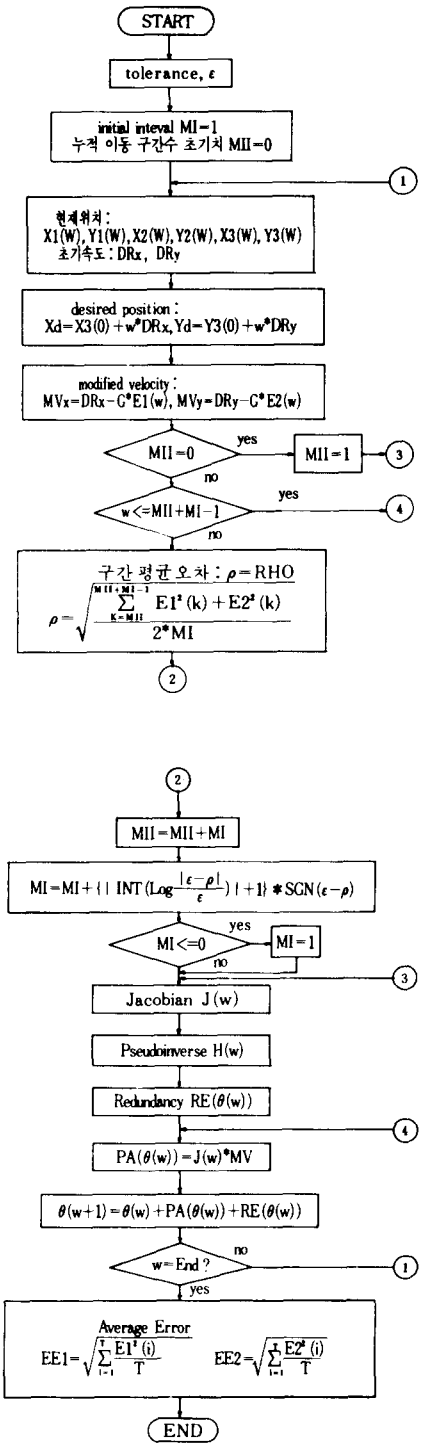

그림 2. 머니퓰레이터를 제어하는 방법에 대한 전체 흐름도.

Fig. 2. Overall flow chart for method of manipulator control.

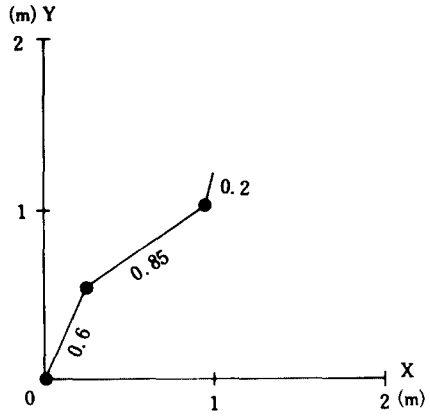

그림 3. 3관절 머니퓰레이터
Fig. 3. 3 −link manipulator.

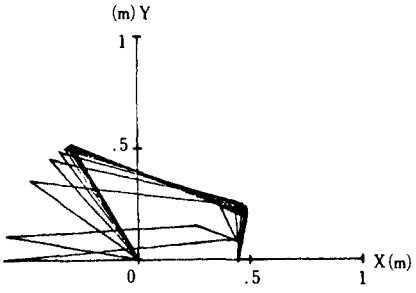

그림 4. 시뮬레이션 결과.
Fig. 4. Simulation results.

표 1. 평균 오차와 Jacobian 행렬의 계산수. 오차의 단위 m.
Table. 1. Average errors and the number of computating the jacobian matrix.

	Method1 N=100	m(i+1)−m (i)=Cons. (4)	Proposed ε=5E-4	Proposed ε=1E-4	Proposed ε=5E-5	Proposed ε=1E-5
AE(X)	5.05E-5	1.85E-4	2.19E-4	1.79E-4	1.07E-4	6.49E-5
AE(Y)	5.16E-5	2.54E-4	4.11E-5	2.73E-4	1.55E-4	1.12E-4
계산횟수	100	25	12	17	25	43

ε 오차의 단위 m

는 자코비안 행렬의 계산횟수를 나타내었다. AE(X)와 AE(Y)는 각각 X좌표와 Y좌표의 평균 경로 오차량이다. 첫번째(method. 1, N=100)는 구간을 100등분 하여서 매 구간 J및 J^+를 구한 결과이고 두번째(m(i+1)−m(i)=cons.)는 100구간에서 J 및 J^+를 4구간 이동시 마다 구한 결과이며 뒤의 4가지 결과는 각각의 허용 오차에 대한 제안 방법의 결과이다. Table을 비교하여 볼 때 제안 방법에서의 결과가 효율적 이면서도 오차가 그다지 크지 않음을 보여준다.

시뮬레이션 결과를 종합하여 본 결과, 매 회 J 및 J^+를 구할 경우 오차를 아주 적게까지 할 수는 있었으나 상대적으로 계산상의 효율성은 낮은 결과를 보였다. 일정 구간마다 자코비안 행렬을 구하는 방법은 정밀도는 조금 떨어지나 계산상의 효율성을 높일 수 있었다. 그러나 한번 주어진 일정한 주기마다 자코비안 행렬을 구하므로 오차량이 상당히 줄어든 경우에도 같은 주기를 유지하여야 하는 비효율성이 나타난다. 제안 방법에서는 이러한 점을 보완하여 오차량이 줄어든 경우에는 이에 대응하여 계산의 효율성을 더 높이고 오차량이 증가 하는 경우에는 계산 구간을 줄임으로써 정밀도를 보상하는, 오차량과 계산 효율성을 절충시키는 결과를 보았다.

Ⅵ. 결 론

본 논문에서는 여유 자유도를 갖는 로보트를 역기구학적으로 제어하기 위하여 수치 알고리즘을 이용하였다. 기구학적인 방정식을 푸는 경우에 필요한 자코비안 행렬 및 가상 역 행렬을 일반적이고 효율적인 방법으로 구하는 알고리즘을 구현하였으며 경로 오차량과 계산 효율성을 두 가지 다 고려하여 절충시키는 방법을 제안하였다.

시뮬레이션 결과, 본 방법은 경우에 따라서는 두 조건 모두 만족할 수 있는 결과를 나타내었다. 따라서 다관절 머니퓰레이터의 경우 계산량을 많이 줄일 수 있음으로 실시간 경로 제어시에 사용이 가능하리라 생각된다.

參 考 文 献

[1] T. Yoshikawa, "Analysis and Control of Robot Manipulators with Redundancy," The First International Symposium of Robotics Research, pp. 726-734, 1984.

[2] H. Hanafusa, "Analysis and control of articulated robot arms with redundancy," *Prep. 8th IFAC World Congress, XIV*, pp. 78-83, Aug. 1981.

[3] A.A. Maciejewski and C.A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying

environments," *The International Journal of Robotics Research,* vol. 4, no. 3, pp. 109-117, Fall, 1985.

[4] A.A. Goldenbug, "A complete generalized solution to the inverse kinematics of Robots," *IEEE Journal of Robotics and Automation,* vol. RA-1, no. 1, pp. 14-20, 1985.

[5] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanics," *IEEE Trans. on Syst. Man and Cybern., SMC-7,* no. 12, pp. 868-871, December 1977.

[6] J. Furusho and S. Onishi, "An efficient approach for solving the inverse kinematics of manipulators," *15th ISIR,* vol. 2, pp. 1051-1058, 1985.

[7] S. Barnett, Matrices in Control Theory with Applications to Linear Programming, pp. 130-136, Van Nostrand Reinhold Company, London, 1971.

[8] J.J. Craig, Introduction to Robotics: Mechanics and Control, Addison Wesley, 1986.

[9] R.P. Paul, Robot Manipulators: Mathematics, Programming and Control, The MIT Press, Cambridge, Mass., 1981.

[10] L.T. Wang and B . Ravani , "Recursive computation of kinematic and dynamic equations for mechanical manipulators," *IEEE Journal of Robotics and Automation,* vol. RA-1, no. 3, pp. 124-131, 1985, 9.

[11] D.E. Orin and W.W. Schrader, "Efficient Jacobian determination for robot manipulators," The First International Symposium of Robotics Research, pp. 726-734, 1984.

[12] D.E. Whitney, "Mathematics of coordinated control of prosthetic arms and manipulators," *Trans. ASME Ser. G, 94,* pp. 303-309, 1972.