'93 한국자동제어학술회의논문집 (1993. 10. 20~22)

3차원 그래픽을 이용한 오프-라인 프로그램의 개발

°박민조, 손 권, 안두성, 이면형
부산대학교 공과대학 기계공학과

A Study on Off-Line Programming Using Three-Dimensional Graphics

°M.J. Park, K. Son, D.S. Ann, and M.H. Lee
Pusan National University

ABSTRACT

The role of a robot becomes more important as factory automation is widely spread in the manufacturing industry. An off-line program system has been required for uninterruption of production lines because it can save cost and time spent in adjusting a robot to a new workcell. The objective of this paper is to develop our own OLP system for a SCARA type FARA robot with four axes. Three-dimensional graphic results are presented for the case when the robot is simulated using the computed torque method with a PD controller and the continuous path trajectory planning.

1. 서 론

생산성 향상을 위해 공장자동화를 추구하는 과정에서 로보트의 도입이 급증하고 있다. 이와 함께 시장변화에 따른 제품의 다양화 추세로 작업 종류와 환경이 빈번히 바뀜에 따라 로보트 시스템의 유연성에 대한 요구도 증대되고 있다. 작업경로의 변경에 따른 작업교시를 위해서 온-라인(on-line) 방식의 경우 전 생산라인의 작업을 중단시켜야 하므로 많은 시간과 비용이 소모된다. 이러한 문제점은 실제작업과 유사한 환경을 가진 시뮬레이터, 즉 오프-라인 프로그래밍(off-line programming, OLP)을 이용함으로써 해결될 수 있다[1]. OLP를 이용하면 로보트의 가동 중에도 로보트의 동적시뮬레이션이 가능하며, 작업교시, 궤적계획, 제어알고리즘의 개발 및 성능평가 등을 소프트웨어를 통해 수행할 수 있다[2,3].

본 논문에서는 SCARA형 로보트에 대한 3차원 그래픽 동적시뮬레이터인 OLP를 개발하였다. 해석대상으로 4축 SCARA형 FARA로보트를 택하였으며, 궤적은 연속경로법에 의해 계획되었고, 제어방법은 계산토크법(computed torque method)에 의한 PD 비선형 보상제어를 적용하였다. 개발된 OLP는 C언어를 이용하여 사용자 인테페이스가 가능하도록 프로그램되었으며, 마이크로프로세서(AT 486) 에서 수행이 가능하고, 로보트 거동에 대한 3차원 그래픽 애니메이션이 가능하며, 시뮬레이션한 궤적에 대한 평가를 할 수 있는 기능을 갖추었다.

2. 3차원 그래픽구현

2차원 화면을 현실감 있는 3차원 모델로 표현하기 위한 개념도는 Fig. 1과 같다. 그림에서와 같은 단계로 좌표변환을 이용하면 원하는 스크린상의 3차원 이미지구현이 가능해진다[4].

2.1 실제좌표계에서 시각좌표계로의 변환

물체를 화면에 3차원으로 그리기 위해서는 물체의 표현기준을 설정해야만 한다. 본 연구에서는 표현기법을 wire-frame모델로 택하였기 때문에 물체의 꼭지점을 표현기준점으로 정하였다.

화면상의 3차원 표현을 위해서는 관찰자의 눈이 시각좌표계의 원점에 있다고 간주하여 관찰자의 입장 즉 시각좌표계에서의 물체의 좌표값을 구하여야

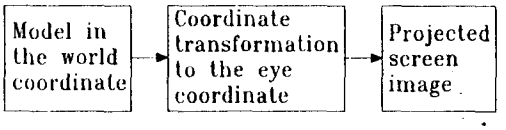

Fig. 1 Process for three-dimensional graphics

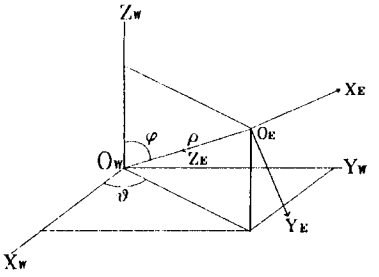

Fig. 2 World and eye coordinates

한다. 실제좌표계(X_W, Y_W, Z_W)와 시각좌표계(X_E, Y_E, Z_E) 사이의 관계는 Fig. 2와 같다. Fig. 2에서 관찰자의 시선방향은 Z_E축을 향하고 O_W와 O_E사이에 화면이 있다고 간주한다.

좌표변환을 위한 동차변환행렬 A는 4×4의 행렬로 실제좌표계 벡타 [V_W]를 시각좌표계 벡타 [V_E]로 변환하기 위한 식은 다음과 같다.

$$[V_E] = A\,[V_W] \tag{1}$$

여기서 변환행렬 A는 아래와 같다.

$$A=\begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 \\ \cos\theta\cos\phi & \sin\theta\cos\phi & -\sin\phi & 0 \\ -\cos\theta\sin\phi & -\sin\theta\sin\phi & -\cos\phi & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

식 (2)에서 변수 θ, φ, ρ를 이용하여 관찰자의 위치를 변경할 수 있다.

2.2 3차원 표현을 위한 원근투영

시각좌표계로 변환된 물체를 화면상에 현실감있게 나타내려면 원근투영이 필요하다. 원근투영의 원리를 도시한 Fig. 3에서 화면상의 좌표값 SX와 SY를 구하면 다음과 같다.

$$SX=\frac{D}{Z_E}\cdot X_E+CMAX_X/2$$

$$SY=\frac{D}{Z_E}\cdot Y_E+CMAX_Y/2$$

여기서 CMAXx, CMAXy는 각각 화면에서 수평축과 수직축의 최대 픽셀수이며, VGA모니터의 경우는 각각 640과 480이다.

실제 프로그램에서도 위의 값들을 변수로 하는 메뉴를 포함하고 있다. 즉 θ, φ, ρ, D를 변수로 두고 원하는 화면에 맞게 사용자가 임의로 조정할 수 있다. Fig. 4는 실제 프로그램상에서의 화면 예이다. 여기서 사용자는 키보드나 마우스를 이용하여 원하는 위치에 가상의 카메라를 둠으로써 시각좌표계의 위치를 변경할 수 있다.

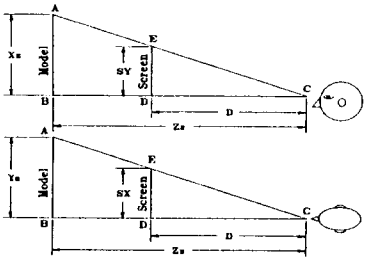

Fig. 3 The perspective projection changing eye coord. to 2-dim. screen coord.

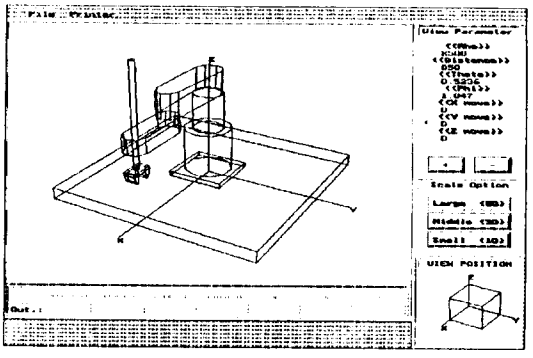

Fig. 4 Example of VIEW menu

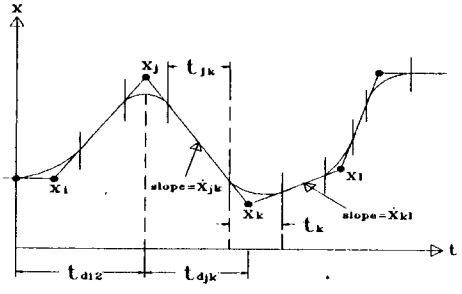

Fig. 5 Multisegment linear path with blends

3. 궤적 계획

동적 시뮬레이션을 위한 궤적계획으로 본 연구에서는 연속경로법을 사용하였다. 이 방법은 실제좌표계에서 경로를 직선화하는 것이며, 운동의 처음과 마지막에서 속도에 불연속성이 발생하나 각 점에 포물선의 혼합구간영역을 첨가함으로써 속도의 연속성을 얻을 수 있다. Fig. 5는 위치가 x_i, x_j, x_k, x_l인 여러 경로점에 대해 혼합구간영역을 나타낸 그림이다. 그림에서 t_k는 경로점 x_k에 있는 혼합영역에서의 시간을 나타내며, t_{jk}는 x_j와 x_k사이의 직선구간에서의 시간을 나타낸다.

4. 정기구학과 역기구학

Denavit-Hartenberg(D-H) 표시법과 정기구학을 이용하여 기준 좌표계에 대한 i번째 링크의 위치와 방향을 표시하는 동차변환행렬은 다음과 같이 구할 수 있다[5].

$$A_j^0 = \prod_{i=1}^{j} (A_i^{i-1}) \qquad (3)$$

여기서

$$A_i^{i-1} = \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i S\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$(C\theta_i = \cos\theta_i, \quad S\theta_i = \sin\theta_i)$

본 연구에서는 Fig. 6과 같은 4링크 SCARA형 로보트를 고려하였다. 단말효과기(end-effector)의 기준좌표계에 대한 동차변환행렬은 식 (3)을 이용하여 구할 수 있다. D-H 표시법에 의한 좌표계에 대한 파라메타는 Table 1과 같다. 표를 이용하여 SCARA형 로보트의 동차변환행렬은 다음과 같다.

$$A_4^0 = \begin{bmatrix} C\theta_{1+2-4} & S\theta_{1+2-4} & 0 & L_1C\theta_1 + L_2C\theta_{1+2} \\ S\theta_{1+2-4} & -C\theta_{1+2-4} & 0 & L_1S\theta_1 + L_2S\theta_{1+2} \\ 0 & 0 & -1 & d_1 + d_3 - d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$(C\theta i = \cos\theta i, \quad S\theta i = \sin\theta i)$

단말효과기의 위치에 대응하는 각 관절각을 구하기 위해서 기하학적인 방법을 사용하였다. Fig. 7

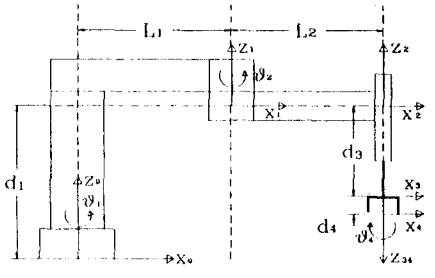

Fig. 6 D-H coordinate frame assignment for SCARA robot

Table 1 Parameters of SCARA robot by D-H representation

Joint	θ_i	α_i	a_i	d_i
1	θ_1	0	L_1	d_1
2	θ_2	0	L_2	0
3	0	180°	0	d_3
4	θ_4	0	0	d_4

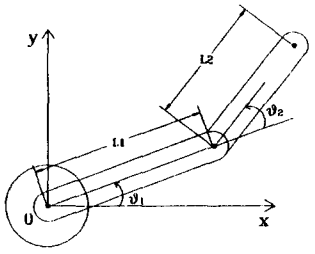

Fig. 7 Coordinate system of two-link SCARA robot

에 도시된 기준좌표계에 대한 단말효과기의 위치로부터 관절각 θ_1과 θ_2의 관계식을 구하면 다음과 같다.

$$\cos\theta_2 = \frac{x^2+y^2-L_1^2-L_2^2}{2L_1L_2} = d$$

$$\theta_2 = \tan^{-1}\left(\frac{\pm\sqrt{1-d^2}}{d}\right) \qquad (4.a)$$

$$\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \theta^* \qquad (4.b)$$

여기서 $\theta^* = \tan^{-1}\dfrac{L_2\sin\theta_2}{L_1+L_2\cos\theta_2}$ 이다.

병진운동의 변수인 d_3과 θ_4는 Fig. 6에서 쉽게 구할 수 있으며 식은 다음과 같다.

$$d_3 = d_1 - z \qquad (4.c)$$

$$\theta_4 = \theta_1 + \theta_2 - \alpha \qquad (4.d)$$

여기서 α은 실제좌표계에서 결정되어지는 단말효과기의 각도를 의미한다.

5. 동적 제어 시뮬레이션

SCARA형 로보트는 세개의 회전관절과 한개의 병진관절로 구성되어있다. 매니퓰레이터를 강성체로 가정하여 라그랑지 방법으로 각 링크의 운동방정식을 유도하면 다음과 같다[6].

$$\begin{aligned} &H_{11}\ddot{\theta}_1 + H_{12}\ddot{\theta}_2 + H_{14}\ddot{\theta}_4 + h_{112}\dot{\theta}_1\dot{\theta}_2 + h_{122}\dot{\theta}_2^2 = \tau_1 \\ &H_{21}\ddot{\theta}_1 + H_{22}\ddot{\theta}_2 + H_{24}\ddot{\theta}_4 + h_{211}\dot{\theta}_1^2 = \tau_2 \\ &H_{33}\ddot{\theta}_3 + G_3 = \tau_3 \\ &H_{41}\ddot{\theta}_1 + H_{42}\ddot{\theta}_2 + H_{44}\ddot{\theta}_4 = \tau_4 \end{aligned} \qquad (5)$$

여기서 매니퓰레이터의 관성행렬(4×4)은 다음과 같다.

$$H = \begin{bmatrix} H_{11} & H_{12} & H_{13} & H_{14} \\ H_{21} & H_{22} & H_{23} & H_{24} \\ H_{31} & H_{32} & H_{33} & H_{34} \\ H_{41} & H_{42} & H_{43} & H_{44} \end{bmatrix}$$

$$H_{11} = m_1 L_{c1}^2 + I_1 + m_2(L_1^2 + L_{c2}^2 + 2L_1 L_{c2}\cos\theta_2) + I_2 + (m_3+m_4)(L_1^2 + L_2^2 + 2L_1 L_2\cos\theta_2) + I_4,$$
$$H_{12} = H_{21} = m_2(L_{c2}^2 + L_1 L_{c2}\cos\theta_2) + I_2 + (m_3+m_4)(L_2^2 + 2L_1 L_2\cos\theta_2) + I_4,$$
$$H_{22} = m_2 L_{c2}^2 + I_2 + m_3 L_2^2 + m_4 L_2^2 + I_4,$$
$$H_{14} = H_{41} = H_{24} = H_{42} = -I_4,$$
$$H_{33} = m_3 + m_4,$$
$$H_{44} = I_4,$$
$$H_{13} = H_{31} = H_{23} = H_{32} = 0.$$

코리올리력과 원심력은 다음과 같이 유도된다.

$$h_{ijk} = \frac{\partial H_{ij}}{\partial q_k} - 0.5\frac{\partial H_{jk}}{\partial q_i}$$

여기서

$$h_{112} = -2m_2 L_1 L_{c2}\sin\theta_2 - 2(m_3+m_4)L_1 L_2\sin\theta_2,$$
$$h_{122} = -m_2 L_1 L_{c2}\sin\theta_2 - (m_3+m_4)L_1 L_2\sin\theta_2,$$
$$h_{211} = m_2 L_1 L_{c2}\sin\theta_2 + (m_3+m_4)L_1 L_2\sin\theta_2,$$
$$h_{212} = -0.5(m_2 L_1 L_{c2}\sin\theta_2 + (m_3+m_4)L_1 L_2\sin\theta_2) = -h_{221}.$$

G_i는 중력을 나타내는 항으로 다음과 같이 정의된다.

$$G_i = \sum_{i=1}^{4} m_j g^T J_i^{(j)}$$

여기서

$$G_1 = G_2 = G_4 = 0,$$
$$G_3 = -g(m_3+m_4).$$

계산토크법을 사용한 제어알고리즘의 블록선도는 Fig. 8과 같다. 그림에서 번호 1, 2, 3은 각각 관성모멘트, 코리올리력, 중력항의 보상을 의미한다. 입력으로는 각 시간에 대한 각 관절의 각도, 각속도, 각가속도이며, 이 값들은 궤적계획과 기구학에 의해 구해진다.

제어기로는 비선형을 보상한 비례 미분제어기를 사용하였으며, 이 때 제어입력은 다음과 같다.

$$\tau(t) = H_a(q)(\ddot{q}^d(t) + K_v[\dot{q}^d(t) - \dot{q}(t)] + K_p[q^d(t) - q(t)]) + h_a(q,\dot{q}) + G_a(q) \quad (6)$$

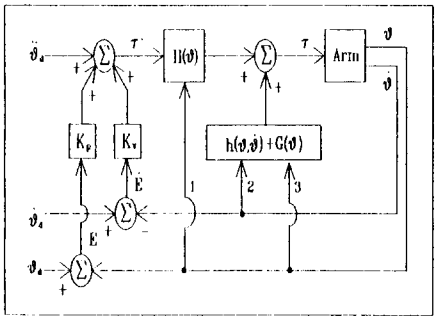

Fig. 8 Block diagram of control system

식 (6)을 운동방정식에 대입하면 다음과 같다.

$$H(q)\ddot{q} + h(q,\dot{q}) + G(q) = H_a(q)(\ddot{q}^d(t) + K_v[\dot{q}^d(t) - \dot{q}(t)] + K_p[q^d(t) - q(t)]) + h_a(q,\dot{q}) + G_a(q) \quad (7)$$

식 (7)는 관성력, 코리올리력, 중력 등을 모두 보상한 경우에 해당되며, 이 식을 정리하면 다음과 같은 오차방정식을 구할 수 있다.

$$H(q)[\ddot{e}(t) + K_v\dot{e}(t) + K_p e(t)] = 0 \quad (8)$$

단, $e(t) = q^d(t) - q(t)$이며, $\dot{e}(t) = \dot{q}^d(t) - \dot{q}(t)$이다. 비선형 연립 미분방정식 (8)의 해는 4위의 Runge-Kutta법을 이용하여 수치적분으로 구하였다[7].

6. OLP의 구성

본 연구의 OLP는 앞의 내용을 토대로 다음의 순서에 따라 진행되며, 기본적인 기능 이외에 화일관리, 편집기 등과 같은 여러가지 부수적인 기능을 가지고 있다.

6.1 궤적계획

궤적계획을 하기 이전에 Fig. 9와 같은 화면의 teaching mode에서 궤적계획을 위한 정보를 얻을 수 있다. 그림에서 각 축에 대한 버튼을 이용하여 단말효과기의 위치를 변경할 수 있으며 그것에 해당하는 정보를 알 수 있다. 이 값들을 이용하여 궤적생성을 하고 화일에 저장하게 된다. 이 정보는 동적 시뮬레이션을 할 경우 입력항으로 이용되며, 다음의 programming mode가 지원하는 시뮬레이션을 순서대로 행할 수 있다.

6.2 동적 제어 및 3차원 그래픽 애니메이션

기본 메뉴로 또는 로보트 프로그래밍 언어의 기초가 되는 인터프리터용 명령어로 궤적계획에서 얻은 값을 이용하여 동적 시뮬레이션이 가능하며, 그

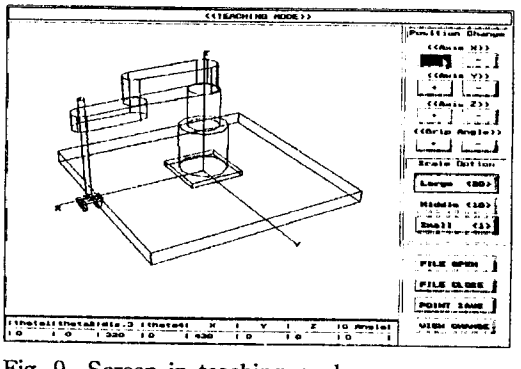
Fig. 9 Screen in teaching mode

결과를 3차원 그래픽 애니메이션으로 로보트의 거동을 볼 수 있다. 예를 들어 "DISPLAY"와 같은 기본메뉴와 "MOVE"와 같은 단말효과기를 이동시키는 명령어가 있다.

6.3 시뮬레이션 결과의 평가

관절공간에서의 각 링크의 궤적과 속력을 비교평가할 수 있는 기능과 실제좌표계에서의 단말효과기의 궤적을 평가할 수 있는 기능을 가지고 있으며

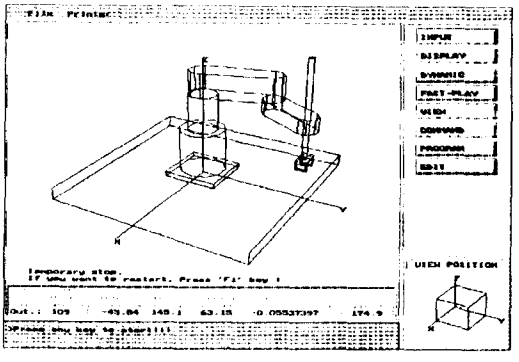

Fig. 10 Screen in programming mode

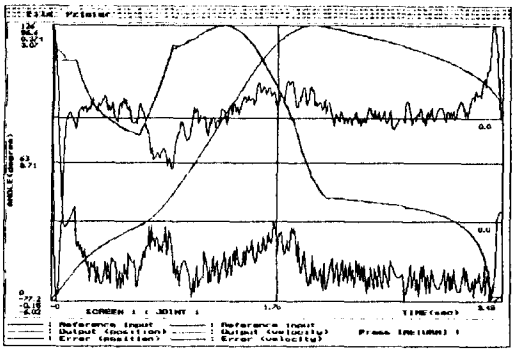

Fig. 11 Screen of performance evaluation in joint coordinate

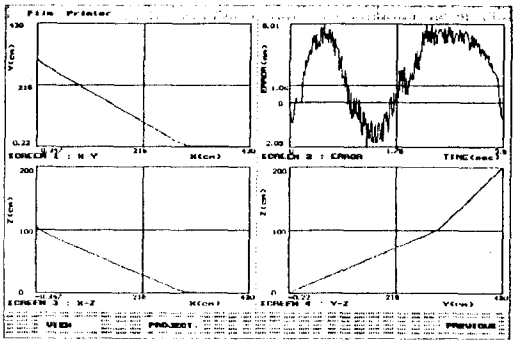

Fig. 12 Screen of performance evaluation in world coordinate

Fig. 11은 관절공간에서의, Fig. 12는 실제좌표계에서의 성능을 나타내는 화면을 나타낸다.

7. 결 론

본 논문에서는 4축 SCARA형 로보트에 대한 오프-라인 프로그래밍의 연구를 행하였으며, 3차원 그래픽 애니메이션, 기구학, 궤적계획, 동역학, 제어알고리즘 등을 이용하여 한 프로그램 안에서 로보트의 작동과정을 시뮬레이션할 수 있음을 보였다. 시뮬레이션 결과에 대해 사용자가 분석할 수 있는 기능을 소프트웨어에 추가시킴으로써 로보트의 사양, 작업환경, 제어알고리즘 등이 변경될 때 로보트의 거동을 확인할 수 있는 OLP를 개발하였다. 또한 OLP가 보다 현실에 가까운 task level상의 프로그램이 가능하도록 온선 처리, 충돌검색 등을 포함한 제어용 언어를 사용하는 OLP의 구현을 위해 연구를 계속 수행하면서 현재의 프로그램을 개선해 나갈 것이다.

8. 참고문헌

[1] 전향식, 최영규, "개인용 컴퓨터 그래픽스를 사용한 로보트 시뮬레이터의 연구," 부산대학교 공과대학 논문집, 제43집, pp. 119~126, 1992.

[2] J. J. Craig, "Issues in the Design of Off-Line Programming Systems," Int. Sym. of Robotics Res., Cambridge, 1988.

[3] J. J. Craig, Introduction to Robotics Mechanics and Control. New York: Addison-Wesley, 1989.

[4] Chan S. Park, Interactive Microcomputer Graphics. New York: Addison-Wesley, 1985.

[5] K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, ROBOTICS: Control, Sensing, Vision and Intelligence. New York: McGraw-Hill, 1987.

[6] H. Asada and J. J. E. Slotine, Robot Analysis and Control. New York: John Wiley & Sons, 1986.

[7] M. L. James, G. M. Smith, and J. C. Wolford, Applied Numerical Methods for Digital Computation. New York: Harper Collins, 1993.