개인용 컴퓨터 그래픽스를 사용한 로보트 시뮬레이터의 연구

田 香 植*○ · 崔 榮 奎**

*대우중공업(주) 연구원, **부산대학교 전기공학과 부교수

A Study on Robot Simulator by use of Personal Computer Graphics

Hyang-Sig Jun, Young-Kiu Choi

Daewoo Heavy Industries LTD., Pusan National Univerity

Abstract

Recently, robot plays an important role in factory automation and its use is rapidly increasing. In paticular, modern industry tends to need the robot to adjust itself swiftly to new workcell. But, robot is to be taught by operator through teaching box in order that it can carry out new tasks. In the process, interruption of the production line is frequent and so much time and expense are required. Computer Graphic Simulator(CGS) provides a useful tool to overcome such a problem. In this paper, CGS consists of teaching mode and execution mode. The 5-axis Rhino robot is modelled in IBM-PC/386 compatible 32- bit personal computer with 80387 co-processor and the simulator performs defined operations on the computer screen.

1. 서 론

최근에 이르러 생산성 향상의 목적과 유해 작업환경에 있어서는 로보트의 사용이 급증하고 있다. 이와 더불어 시장변화에 따른 제품의 다양화와 제품 수명의 단축 등으로 작업환경과 작업요구가 빈번히 변화하므로 로보트 시스템의 유연성(flexibility) 등이 점점 더 요구되고 있다. 이러한 요구를 충족시키기 위하여 로보트 시스템을 신속하게 설계하고, 로보트의 교시(teaching program) 및 작업배치 등을 위해 소모되는 시간과 비용을 줄일 수 있는 방법에 대한 연구가 필요하게 되었다.

로보트 교시방법 중 작업현장에서 직접 작업을 교시하는 온-라인(on-line)방식은 로보트를 교시하는 동안에는 전 생산라인(production line)의 작업이 중단되어야 하고, 복잡한 작업에는 적용이 쉽지 않다.[1] 이와 같은 종래의 시행착오적인 교시방법을 답습하는 경우, 날로 증가하는 사용자의 다양한 요구조건을 충족시키기 위해서는, 그 설계 및 교시에 소요되는 시간과 비용이 증가한다는 것이 심각하게 지적되고 있다.[2][3][4] 위와 같은 문제점은 실제의 작업현장 상황과 유사한 환경을 가진 컴퓨터 그래픽 시뮬레이터(computer graphic simulator)를 이용함으로써 많은 부분이 해결될 수 있다.[1][5] 그래픽 시뮬레이터는 로보트의 작업환경과 행동반경을 시각화할 수 있으므로, 선택된 로보트 및 주변 물체의 기종이 적합한 지를 확인할 수 있고, 계획된 경로상에서 로보트와 로보트간에 또는 로보트와 주변 물체간에 충돌이 발생하는지, 로보트 관절의 운동범위(joint limit)를 넘지 않는지등을 컴퓨터 화면상에서 확인할 수 있다. 또한 그래픽 시뮬레이터는 교육용 패키지로 사용될 수 있으므로, 저렴한 비용으로 로보트에 대한 교육과 연구에 적극적인 참여를 유도할 수 있다.[6][7] 그래픽 시뮬레이터의 연구에서는 로보트를 구성하는 링크들의 기하학적인 모델링(modelling)방법과 여러 링크들이 어떻게 결합되어 로보트를 구성하는 가에 대한 연구가 수행된 바 있고, 기존에 사용되고 있는 로보트를 이용하고자 하는 경우에 설계시간을 최대한으로 단축하기 위하여 국내외적으로 사용되고 있는 로보트들을 데이타베이스화 하는 연구도 행하여졌다.[5][8][9]

본 논문에서는 볼랜드(Borland)사의 C++언어로 로보트의 작업환경과 행동반경을 시각화 할 수 있는 교육용 로보트의 그래픽 시뮬레이터를 개인용 컴퓨터상에서 구현하여 교시에 소요되는 시간과 비용을 절감할 수 있는 방법을 제시하였고, 또한 이것을 교육용 패키지로도 사용할 수 있다. 본 논문의 구성은 다음과 같다. 2장에서는 교육용 로보트의 각 링크들을 실제 모양과 흡사하면서도 계산량을 감소시킬 수 있는 모델링을 제안했고, 3장에서는 로보트의 3차원 운동을 컴퓨터 화면상에서 시각화하여 작업교시를 할 수 있는 교시 모우드(teaching mode)와 교시 모우드에서 교시된 작업을 직접 수행하여 컴퓨터 화면상에서 확인할 수 있는 실행 모우드(execution mode)를 구현하였다.

2. 로보트 모델링와 상호결합관계

2.1 링크의 모델링

라이노(Rhino) 로보트는 미국의 라이노사의 교육용 로보트로써 5개의 링크와 2개의 지지대로 구성된 5축 로보트이다[10]. 링크를 모델링할 때는 원형에 충실하고 가능한 가장 간단한 형태로 모델링해야 한다. 충실도를 높이기 위하여 원형에 치중하다보면 많은 데이터(data)량과 추가되는 알고리즘(algorithm)들로 인하여 계산량이 증가되는 부담이 있고, 너무 단순한 모델은 조잡해지는 문제점이 있다. 계산량을 줄이면서도 원형의 충실도를 유지할 수 있는 모델링방법이 필요했다. 그 방법으로써 각 링크의 부속장치를 제외하고 각 링크를 각다면체화 했다. 우선 지지대에서부터 단말효과기(end-effector)쪽으로 나가면서 링크번호를 붙인다. 링크1은 4각 기둥화 했으며, 링크 2,3은 약간 변형된 4각기둥으로 모델링했고, 링크 4는 원형에 충실하게 모델링했다. 링크 5는 육면체 4개를 조합하여 구성하였다. 지지대는 육면체로 구성했다.[13]

2.2 링크의 상호관계

라이노 로보트에서 각 링크들의 위치와 방향관계는 데네비트-하텐버그(Denavit-Hartenberg)의 표현방법을 사용하였다. Fig. 1에는 본 논문에서 모델링한 링크들과 지지대로 구성한 라이노 로보트의 각 링크 좌표계가 표시되어 있고, Table 1에는 링크 좌표계에서 설정된 라이노 로보트의 기하학적 링크변수들이 주어져 있다.

Table 1 Rhino robot arm link coordinate parameters

joint i	θ_i	α_i	d_i	a_i
1	θ_1 = 90°	-90°	0	0
2	θ_2 = -90°	0	0	a_2 = 250
3	θ_3 = 90°	0	0	a_3 = 250
4	θ_4 = 180°	90°	0	0
5	θ_5 = 0	0	0	0

θ_i: 각 링크의 회전각 a_i: 각 링크의 길이
α_i: 링크간의 오프셋 각 d_i: 링크좌표계의 원점간의 거리

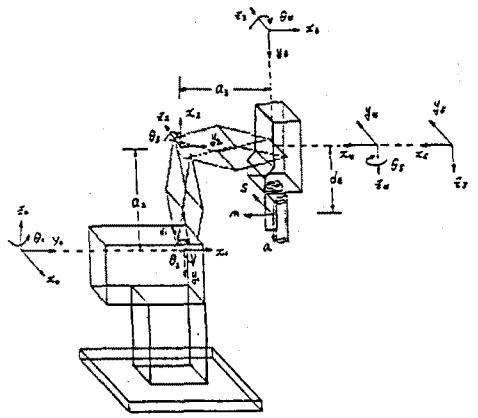

Fig. 1 Link coordinate systems for Rhino robot

기준 좌표계에 대한 j번째 링크의 위치와 방향을 표시하는 등차 변환 행렬(homogeneous transformation matrix) ^0A_j는 다음과 같이 표시된다.

$$^0A_j = \prod_{i=1}^{j} (\ ^{i-1}A_i\ ) \qquad (\ j = 1,2,3,4,5\ ) \qquad (\ 1\ )$$

여기서 행렬 $^{i-1}A_i$는 i-1번째 링크 좌표계에 대한 i번째 링크 좌표계의 위치와 방향을 나타내는 등차 변환 행렬[11]이다. 그리고 Table 1의 변수들을 이용하여 베이스에 대한 단말효과기(end-effector)의 등차 변환 행렬 ^0A_e를 구하면 다음과 같다.

$$^0A_e = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (\ 2\ )$$

$n_x = C_1C_{234}C_5 - S_1S_5$, $n_y = S_1C_{234}C_5 + C_1S_5$, $n_z = -S_{234}C_5$

$s_x = -C_1C_{234}S_5 - S_1C_5$, $s_y = -S_1C_{234}S_5 + C_1C_5$, $s_z = S_{234}S_5$

$a_x = C_1S_{234}$, $a_y = S_1S_{234}$, $a_z = C_{234}$

$p_x = d_e(C_1S_{234}) + a_3C_1C_{23} + a_2C_1C_2$

$p_y = d_e(S_1S_{234}) + a_3S_1C_{23} + a_2C_2S_1$

$P_z = d_eC_{234} - a_3S_{23} - a_2S_2$

윗 식에서 $S_i = \sin\theta_i$, $C_i = \cos\theta_i$, $S_{ij} = \sin(\theta_i+\theta_j)$, $C_{ij}=\cos(\theta_i+\theta_j)$, $S_{ijk}=\sin(\theta_i+\theta_j+\theta_k)$, $C_{ijk}=\cos(\theta_i+\theta_j+\theta_k)$, (i,j,k = 1,2,...,5)를 사용하였고, d_e는 5번째 관절 좌표계의 원점과 단말효과기의 좌표계의 원점간의 거리이다.

3. 그래픽 시뮬레이터

3.1 그래픽 시뮬레이터의 구조

본 절은 앞에서 모델링한 링크로 구성된 로보트의 움직임을 시각화할 수 있는 그래픽 시뮬레이터를 구현하였다. 여기서 로보트 그래픽 시뮬레이터는 교시 모우드와 실행 모우드로 나누어진다. 교시 모우드에서는 실제의 교시 박스(teaching box)를 컴퓨터 키보드의 특정한 키로 대체했다. 실행 모우드에서는 교시 모우드에서 교시된 작업을 계획된 경로를 따라 수행하는 것을 컴퓨터 화면상에서 확인할 수 있도록 했다. 그래픽 시뮬레이터의 프로그램 순서도는 Fig. 2와 같다.

3.2 교시 모우드(teaching mode)

화면상에 도시되는 라이노 로보트의 크기는 실제 크기의 1cm당 컴퓨터의 픽셀(pixcell) 4개로 잡았다. 그리고 각 꼭지점의 좌표는 그 링크 좌표계에서 초기화했다. 각 꼭지점의 데이터의 입력순서는 윗면을 우선하여 반시계 방향으로 입력했고, 밑면도 같은 방법으로 꼭지점의 좌표를 입력했다. 그리고 각 링크의 회전과 이동은 기준 좌표계에 대한 링크 좌표계의 등차 변환 행렬을 이용하였다. 교시 모우드에서는 컴퓨터 키보드의 일정한 키를 누름으로써 실제 교시 박스에서와 같이 작업을 교시할 수 있도록 했다. 실행 결과는 Fig. 3과 같다.

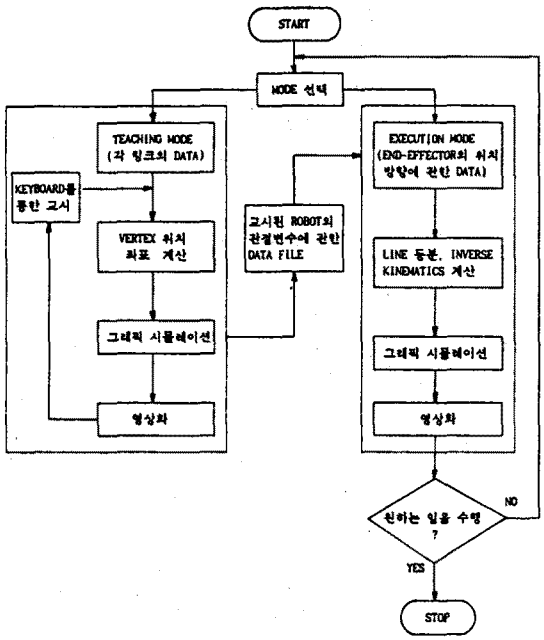

Fig.2 Flowchart of graphic simulator

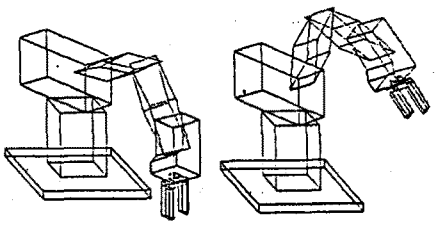

Fig. 3 Link motion of Rhino robot

3.3 실행모우드(execution mode)

실행 모우드에서는 교시 모우드에서 설정된 기준 경로에 단말효과기의 위치와 방향에 해당하는 각 링크의 회전각을 역기구학적인 방법(inverse kinematics)에 의해 구하여 컴퓨터 화면상에서 교시된 작업을 직접 수행하도록 했다. 본 논문에서는 각 링크의 회전각 $\theta = (\ \theta_1, \theta_2, \theta_3, \theta_4, \theta_5)^T$을 기하학적인 접근 방법[12]으로 구하였다.
여기서 $P_3 = (P_{x3}, P_{y3}, P_{z3})^T$는 베이스에 대한 3번째 링크좌표계 원점의 위치벡터라고 하면, 다음 관계식이 성립한다.[13]

$$\theta_1 = \frac{\pi}{2} + \tan^{-1}\left[-\frac{P_{x3}}{P_{y3}} \right],\ \theta_2 = -(\ \alpha + \beta\ ) \qquad (3)$$

$\theta_3 = \pi - \gamma$, $\theta_4 = \pi +$

$$\tan^{-1} \frac{(C1*C2*C3-C1*S2*S3)*a_x+(C2*C3*S1-S1*S2*S3)*a_y-S23*a_z}{(C1*C2*C3+C1*C3*S2)*a_x+(C2*S1*S3+C3*S1*S2)*a_y+C23*a_z}$$

$$\theta_5 = \tan^{-1}\left[\frac{-S1*n_x + C1*n_y}{-S1*s_x + C1*s_y} \right]$$

$$\text{단, } \cos\alpha = \frac{a_2^2+P_{x3}^2+P_{y3}^2+P_{z3}^2-a_3^2}{2*a_2*(P_{x3}^2+P_{y3}^2+P_{z3}^2)},\quad \beta=\tan^{-1}\left[\frac{P_{z3}}{P_{x3}^2+P_{y3}^2}\right]$$

$$\cos\gamma = \frac{a_2^2+a_3^2-(P_{x3}^2+P_{y3}^2+P_{z3}^2)}{2*a_2*a_3}$$ 을 의미한다.

기준 경로 계획은 Fig. 4와 같이 물체를 작업대로 운반하는 일을 로보트에게 지시하고자 한다면, 첫번째 해야 할 일은 물체가 위치한 곳에서부터 작업대까지 로보트가 움직이는 경로를 정하는 것이다. Fig. 4에서 그 경로를 A-B-C-D로 정하고 있다. 그 다음에는 경로를 A-B, B-C, C-D로 구분한 후 각 경로마다 일정한 크기로 등분한다. 등분된 각좌표점에 해당하는 단말효과기의 위치와 방향을 식(3)의 역기구학적인 방법을 이용하여 각 링크의 관절 변수를 계산한다. 계산된 값들을 그래픽 시뮬레이션 프로그램에 입력하여 로보트를 그 좌표점에 위치시킬 수 있다. 위와 같은 과정을 연속적으로 반복하게 되면 로보트는 목적한 D지점까지 물체를 운반하게 된다. 그 실행결과는 Fig. 5에 나타냈다.

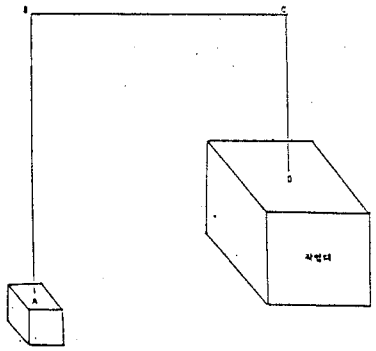

Fig. 4 Path of task

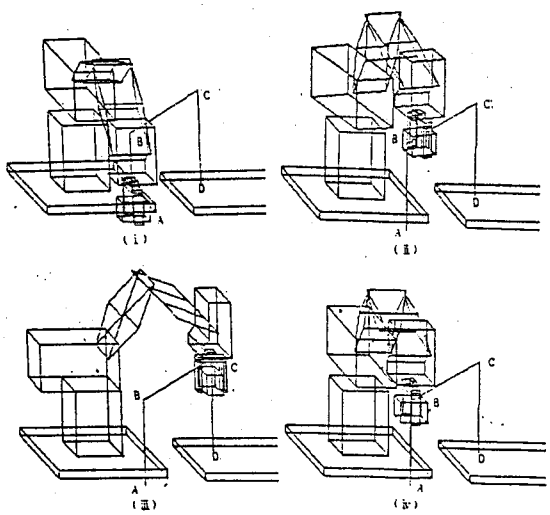

Fig. 5 Rhino robot moving a object from point A to D

3.4 고찰

본 논문에서 실행한 그래픽 시뮬레이터는 MS-DOS지원하에서 보조연산기(coprocesser)가 내장된 32비트(bit) IBM-PC/386을 이용하여 볼랜드(Borland)사의 C++로 구현하였다. 교시 모우드의 소스 파일(source file)의 크기는 26k 바이트(byte)이고 보조 파일의 크기는 14k 바이트이며, 실행 모우드의 소스 파일의 크기는 15k 바이트이고 보조 파일의 크기는 15k 바이트이다. 그래서 그래픽 시뮬레이터의 전체 크기는 60k 바이트 정도이다. 그리고 교시 모우드에서 교시된 각 링크의 관절변수를 저장하기 위하여 데이타 파일(data file)이 필요했다. 영상 속도는 이전 그림을 지우고, 새로운 그림을 그리는데 0.3초 정도가 소요되어 비교적 연속적인 동작을 볼 수 있었다.

4. 결론

본 논문에서는 라이노 로보트의 각 링크의 원형에 충실하면서도 간결한 모델을 제안했고, 모델링된 링크로 라이노 로보트를 구성하여 그래픽 시뮬레이터를 설계했다. 그래픽 시뮬레이터에서는 교시모우드와 실행모우드를 개인용 컴퓨터에서 구현함으로써 로보트의 교시에 소요되는 시간과 비용을 절감할 수 있는 방법을 제시했다. 또한 값비싼 로보트 장비를 구입할 여유가 없는 곳에서 로보트 공학의 기본개념을 예시하기 위해 본 그래픽 시뮬레이터를 교육용 패키지(package)로도 이용할 수 있다.

앞으로는 동력학적인 부분과 제어용언어(control language)의 컴파일러(compiler)를 부가하고 상세한 사용자 메뉴얼을 포함한다면, 원하는 로보트의 설계, 동적 시뮬레이션(dynamic simulation)과 오프-라인 프로그래밍(off-line programming)까지 할 수 있는 그래픽 시뮬레이터를 제작할 수 있을 것으로 사료된다.

참 고 문 헌

1) Rita R. Schreiber, "How to teach a robot," Robotics Today, June (1984)
2) C.C Thomsom, "Robot modelling - the tools needed for for optimal design and utilization," Computer-aided Design, vol. 16 no. 6, Nov.(1984)
3) Kunwoo Lee, "A CAD system for designing robot manipulators," IEEE Internatiotal Conference on Robotics and Automation (1985)
4) Masaharu Takano, "Development of simulation system of robot motion and its role in task planning and design design system," Robotics Research, The 2nd International Symposium (1985)
5) Robot N. Stauffer, "Robot system simulation," Robotics Today, June (1984)
6) T.Sata, F. Kimura and A. Amano, "Robot simulation system as a task programming tool," Proc. of the 11th ISIR (1981)
7) T. Dillmann, B. Hornung, M. Huck and U. "Rembold, interactive programming of robots using textual programming and simulation technique," Proc. of the 16th ISIR (1986)
8) 장원, "컴퓨터 그래픽스를 이용한 로보트 시뮬레이터의 설계," 한국과학기술원, 석사학위논문 (1986)
9) 정명진, 유동상, 고충협, "범용 로보트 시뮬레이터의 개발," 90 로보틱스 및 자동화연구회 Workshop (1990)
10) H.S.Sandhu, Tom Hendrickson, RobotTalk™ User's Manual, Rhino Robots Inc.(1989)
11) Richard P. Paul, Robot manipulators: mathematics, programming, and contol, The MIT Press, pp 41-62 (1979)
12) K.S.Fu, R.C.Gonzalez, C.S.G.Lee, Robotics: control, sensing, vision, and intelligence, McGraw-Hill Book Company, pp 60-76 (1987)
13) 전향식, "개인용 컴퓨터 그래픽스를 사용한 로보트 시뮬레이터의 연구," 부산대학교, 석사학위논문 (1992)