韓國精密工學會 '97年度 秋季學術大會論文集 pp.346~349

Windows NT 환경에서 OpenGL을 이용한 직접구동 협조로봇용 Output Tracking 시뮬레이터 구현

Realization of an output controller simulator based on Windows NT for a direct drive cooperative robot using OpenGL

최 대 범*, 양 연 모*, 안 병 하*

D.B. Choi, Y.M. Yang, B.H. Ahn

Abstract

In this paper we develop a real-time simulator for direct drive cooperative robots by using OpenGL in a Windows NT based system. This simulator is composed of 2 parts, a display part and an interface part. In the display part the robot is modelled and rendered in 3D space. To do this OpenGL , a kind of graphic library, is used for rendering and animating robots and kinematics gives the information of the current robot configuration. The control and the feedback data are sent and received via the interface part. In real time simulation interfacing part needs fast data transfer rate and good noise immunity. In experiment we have simulated 2-link direct drive cooperative robots using the trajectory tracking algorithm proposed in reference.

Key Words : Robot Simulator(로봇 시뮬레이터), OpenGL(그래픽 라이브러리), Kinematics(기구학), Cooperative Robot(협조 로봇)

1.서론

로봇 제어 알고리즘 개발시에 하드웨어 구성이 필요하다. 그러나 로봇을 구성하여 알고리즘을 확인하는 것은 쉬운 일이 아니다. 이를 위해 위험부담이 적고 다루기 편리한 시뮬레이터의 개발이 필요하다. 그리고 현재 PC는 성능은 향상되고 있으며 가격은 내려가고 있다. Windows NT를 운영체계한 시스템은 중간 성능의 워크스테이션 정도의 성능을 가진다. 또한 워크스테이션에서 사용되는 여러 소프트웨어가 Windows NT/95[3]에 이식되어, PC에서의 그래픽 기능이 향상되고 있다. 이러한 상황에서 Windows NT/95를 기반으로 한 로봇 시뮬레이터는 낮은 가격과 최소한의 기능을 가지고 로봇 알고리즘을 개발시 이용할 수 있다.

OpenGL[1,2]은 미국 Silicon Graphics사가 개발한 그래픽 라이브러리로 널리 쓰이고 있다. OpenGL의 역할은 그래픽 하드웨어와의 인터페이스이다. 즉 프로그래머는 OpenGL을 통해 그래픽 하드웨어를 이용하여 그래픽 작업을 한다. 그러나 Windows NT/95는 프로그래머가 직접 하드웨어를 조작할 수 없으며 그래픽 부분에 대해 기본적으로 지원되는 것은 한정되어 있다. 그래서 보다 나은 그래픽 작업을 위해 OpenGL과 같은 그래픽 라이브러리가 Windows NT/95의 운영체계의 한 부분으로 포함되어 나왔다. 이를 이용하여 시뮬레이터를 구성하면 그래픽 부분에서 좋은 효과를 얻을 수 있다.

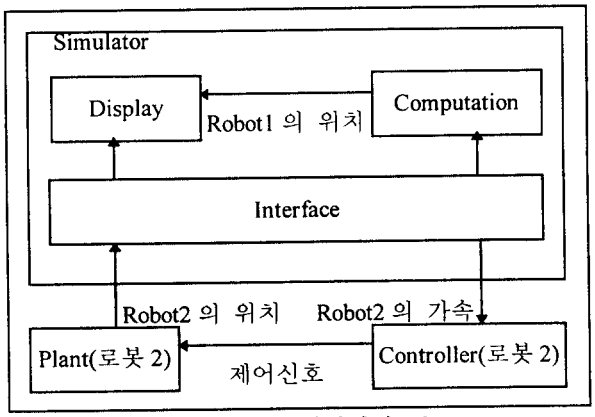
직접 구동(Direct Drive)[7] 로봇은 기어 로봇(Gear Robot)보다 소음이 적고 저속/고속 제어시에 유리하지만 제어기가 복잡하다. 이러한 직접 구동 로봇을 시뮬레이터의 대상으로하여 로봇 시뮬레이터를 이 논문에서 구성하였다. 협조 로봇(Cooperative Robot)[9]의 궤적 추종(Trajectory Tracking)[6,8]이 시뮬레이터의 목적이다.

2. 로봇 시뮬레이터의 구성

2.1 개관

* 광주과학기술원 기전공학과

협조 로봇은 2 대 이상의 로봇이 상호 연계하여 동작하는 로봇이다. 다루고자 하는 부분은 기준(Reference) 로봇의 End-Effector 가 특정 궤적(Trajectory)을 따라 작동할 때, 다른 로봇의 End-Effector 가 이 궤적을 추종하도록 하는 Algorithm[6]을 시뮬레이터를 이용하여 시각적으로 나타내는 것이다. 위의 작업에서 로봇의 시뮬레이터는 기준(Reference) 로봇의 궤적을 만들고 이와 관련된 정보를 다른 로봇에 전달하며 마지막으로 현재의 로봇들의 위치를 3 차원으로 표시한다. 이를 위해 로봇 시뮬레이터는 3 개의 부분으로 나누어진다. 로봇들의 위치를 표시하는 3 차원으로 표시하는 디스플레이 부분, 기준 궤적(Reference Trajectory)과 그와 관련된 정보를 계산하는 부분, 그리고 시뮬레이터와 외부를 연결하는 인터페이스 부분이다. 기준 로봇을 로봇 1, 로봇 1 의 End-Effector 를 추종하는 로봇을 로봇 2 라고 할 때 시뮬레이터를 포함한 협조 로봇 시스템은 그림 1.1 과 같다. 이 장에서는 각 부분을 설명한다.



<그림 1-1 시뮬레이터의 구조>

2.2 Display 부분

로봇들의 위치를 3 차원으로 애니메이션한다. 3 차원 애니메이션을 위해 많은 소프트웨어가 있다. 이 중에서 미국 Silicon Graphics 사에서 나온 OpenGL 을 사용했다.

2.2.1 OpenGL

OpenGL 은 SGI, SUN, DEC 등의 워크스테이션과 Windows NT/95 를 기반으로 한 PC 에서 사용할 수 있는 하드웨어 플랫폼에 독립적인 그래픽 라이브러리이다. Windows NT/95 에서는 운영체계의 한 부분이다. OpenGL 의 특징은 다음과 같이 정리할 수 있다.

OpenGL 의 Primitive 는 10 종류이다. 점(Point), 3 종류의 선(Line), 6 종류의 다각형(Polygon)을 지원한다. 10 가지의 Primitive 를 이용하여 물체를 3 차원으로 모델링한다.

OpenGL 은 Modeling Transformation 과 Viewing Transformation 의 개념을 이용한다. Modeling transformation 은 물체의 각 부분을 정의할 때 필요한 Rotation 과 Translation Transformation 이며, Viewing Transformation 은 View Point 의 위치를 결정할 때 사용되는 Rotation 과 Translation Transformation 이다. 이 2 Transformation 은 4×4 Homogeneous Matrix 로 표현된다. 그리고 2 Matrix 의 곱해 얻어진 Matrix 를 Model-View Matrix 라 하며 OpenGL 은 Rendering 시에 이를 이용한다.

OpenGL 은 Visual Realism 을 높여 주는 여러 함수 및 기능이 있다. 음영(Shading) 효과와 원근 투시법(Perspective View)등은 시각적 효과를 높인다.

2.2.2 로봇 모델링

이 부분에서의 모델링은 역학적인 계산을 위한 것이 아니라 로봇의 외관을 그리는 것으로 OpenGL 의 Model Transformation 부분과 관계가 있다. 이를 위해 우선 로봇을 축(Link)단위의 모듈로 나누었다. 그리고 각각의 모듈들을 OpenGL 의 Cache 인 Display List 에서 모델링하였다. Display List 는 렌더링(Rendering)시에 변화하지 않는 부분을 미리 계산한 후 저장하여 화면에 표시할 때 계산이 없이 쓰이는 부분이다. 즉 로봇의 각 축은 강체이기 때문에 변화하지 않으므로 Display List 를 이용하여 모듈 형태로 사용했다. 각 Display List 안에서 10 가지 Primitive 를 이용하여 로봇의 형태를 표시한다. 이 때 Primitive 들의 종류와 색깔을 이용하여 외부에서 보이는 형태를 근사적으로 나타낸다. 로봇의 음영을 위해 Primitive 에 의해 형성된 평면의 법선 벡터를 계산한다.

2.2.3 렌더링과 애니메이션

모델링된 부분을 화면에 표시하는 것이 렌더링이다. 여기에서는 모듈별로 모델링된 로봇을 조립하여 모니터에 그리는 과정이다. 로봇이 움직임은 Joint Variable 의 변화로 나타내므로 Joint Variable 을 Viewing Transformation 의 인자로 이용하여 로봇의 형태를 그린다. 그외의 View Point 와 Visual Realism 을 위한 부분도 다루어 진다.

애니메이션은 일정 시간을 간격으로 로봇의 형태를 다시 그리는 것이다. 즉 시뮬레이션 이후에 Joint Variable 이 변화하기 때문에 로봇의형태도 바뀐다. 이를 다시 그려주어 로봇을 변화를 연속적으로 나타낸다. 부드러운 애니메이션을 위해 최소한 1

초에 15 frame 을 그려야 하며, 24 frame 정도면 만족할 만하다고 한다. 그러나 애니메이션은 많은 컴퓨터의 자원을 요구하므로 실시간 시뮬레이션에서 많은 부분이 고려되는 부분이다. 로봇 모델링에서 Display List 를 사용한 이유도 불필요하게 반복되는 부분의 계산을 줄여 애니메이션을 효율적으로 하기 위해서 이다.

2.2.3 Windows NT/95 프로그래밍

Windows NT/95 에서 OpenGL 을 사용할 때, 여러 가지 Tool 을 사용할 수 있다. 이 시뮬레이터에서는 Microsoft Visual C++를 이용하였다. Visual C++는 32 비트 윈도우즈 프로그래밍을 위해 개발된 소프트웨어이다. 이 논문에서는 Microsoft 에서 만든 Microsoft Foundation Library(MFC)를 이용했다.

시뮬레이터에서 필요한 부분은 Display 와 통신이다. Display 는 Window 를 만들고, Display 환경을 설정하며, 시뮬레이션 결과를 보여 주는 것으로 MFC 와 OpenGL 을 이용했다. 통신 부분은 시뮬레이터 외부와의 데이터 교환을 하는 부분으로 dSPACE[8]를 이용했다.(dSPACE 는 2.4 인터페이스 부분에서 다룸) dSPACE 에서 제공한 드라이버들과 라이브러리들을 사용하여 시뮬레이터 외부와 통신하는 부분을 만들었다.

Animation 과 통신을 할 때 타이머를 사용해야 한다. 타이머가 일정시간 간격마다 신호를 내보내면 그 때를 기준으로 현재의 로봇을 그리고 입출력 신호를 처리한다. 그러나 Windows NT/95 에서 지원하는 성능은 좋지 않다. Timer 의 Resolution 은 1m 초 이지만 OpenGL 이용할 때 Windows NT 에서는 대략 10m 초, Windows 95 에서는 55m 초 이하의 Timer Resolution 을 얻을 수 없다.

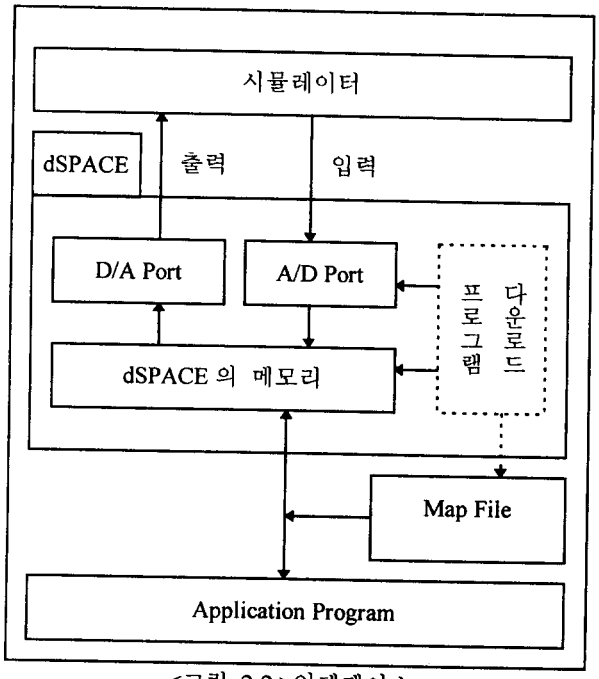
2.3 Computation 부분

기준 궤적을 생성하고 그에 관련된 정보를 계산하는 부분이다. 궤적 구성을 위해 기구학(Kinematics)[7]과 역기구학(Inverse Kinematics)[7]이 필요하다. 궤적은 End-Effector 의 각각의 위치들의 모임이다. 각각의 위치가 결정되면 역기구학을 이용하여 Joint Variable 들을 계산하여 Display 부분으로 보내어 기준 로봇을 3 차원으로 그린다. 다음은 End-Effector 의 전위치와 다음위치를 이용하여 기준 로봇을 추종하는 로봇의 가속도를 계산한다. 이는 기준 로봇을 추종하는 로봇의 제어기로 보낸다.

2.4 인터페이스 부분

시뮬레이터 외부와의 접속을 위해 필요한 부분이다. dSPACE 를 사용하여 A/D 변환과 D/A 변환을 했다. 특히 빠른 변환 속도와 잡음이 적기 때문에 실시간 시뮬레이션에 dSPACE 가 적합하며, 이 논문에서도 사용했다.

dSPACE 를 이용한 인터페이스를 위해 2 가지 작업이 필요하다. 우선 dSPACE 에 다운로드되어 dSPACE 의 A/D D/A 포트를 제어하여 데이터를 처리하는 부분과 dSPACE 의 데이터를 시뮬레이션 프로그램으로 가져오는 부분이다. dSPACE 에서 제공하는 CLIB 이라는 라이브러리를 이용하여 처리할 수 있다. 인터페이스 과정은 다음 <그림 2-2>와 같다.



<그림 2-2 >인터페이스

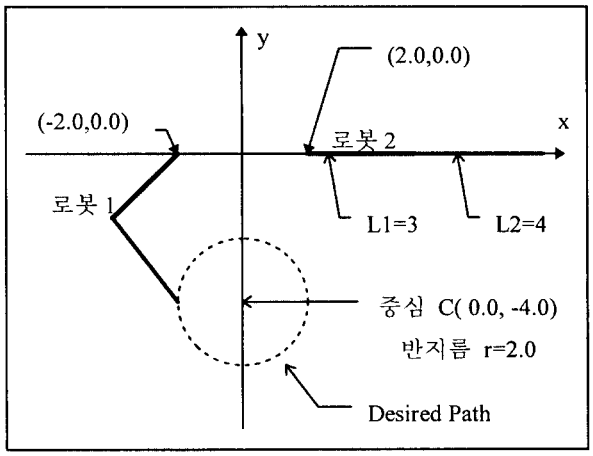
다운로드 프로그램에는 입/출력 데이터를 의한 변수들이 있으며, 이 변수들은 dSPACE 의 메모리에 특정 번지를 가지고 있다. 그리고 다운로드 프로그램을 컴파일할 때에 각 입출력 변수와 기타 사항이 Map file 에 저장된다. 그리고 프로그램이 다운로드되고 데이터가 입출력될 때, 시뮬레이터의 프로그램은 dSPACE 에서 지원하는 CLIB 의 함수와 Map file 을 이용하여 dSPACE 의 메모리를 읽거나 쓴다.

2.5 제어 알고리즘

Trajectory Tracking 알고리즘[6,8]은 Amit Ailon 이 제시한 것을 사용했으며, 제어신호로 Joint variable 의 가속도 정보를 요구한다.

3.실험

시뮬레이션의 대상은 2 대의 2 축 로봇이다. 로봇들의 초기 위치와 Reference Robot 이 생성하는 Desired Trajectory 는 <그림 3-1>과 같다. 로봇 1 은 기준 로봇으로 기준 궤적을 생성하며 움직인다. 임의로 기준 궤적을 원으로 선정했다.



<그림 3-3>초기값과 궤적

로봇 1 이 그리는 궤적은 시뮬레이터에서 생성되며, 이에 따른 가속도 정보가 계산되어 로봇 2 의 제어기로 보내진다. 로봇 2 가 움직이고, 그 결과는 다시 시뮬레이터로 보내진다.

로봇은 100m 초에 한번씩 시뮬레이터와 통신한다. 실험한 결과 로봇 2 의 End-Effector 가 로봇 1 의 End-Effector 를 추종하였다.

4.결론

이 논문에서는 PC 에서 작동하는 2 축용 협조 로봇 시뮬레이터를 구현하였다. PC 는 워크스테이션보다 경제적이고 대중적이기 때문에 PC 기반 로봇 시뮬레이터는 매력적이다. 또한 Windows NT/95 는 사용이 편한 그래픽 사용자 인터페이스(Graphic User Interface, GUI)를 지원하여 사용이 편리하다.

그러나 아직 PC 의 성능이 고성능 워크스테이션에 비해 떨어지고 운영체계인 Windows NT/95 역시 유닉스(UNIX)에 비해 뛰어나지는 않다. 특히 OpenGL 을 이용한 애니메이션을 할 때 타이머를 Windows NT 는 10m 초, Windows 95 는 55m 초 이하로 사용할 수 없었다. 이는 시뮬레이터의 제약이 된다.

앞으로 이 연구를 기초로 하여 3 축용 협조 로봇 시뮬레이터를 구현할 계획이다.

Reference

1. Ron Fosner, "OpenGL Programming for Windows 95 and Windows NT," Addison Wesley Developers Press, Reading Massachusetts, 1996
2. Richard S. Wright Jr. and Micheal Sweet, "OpenGL Super Bible," Waite Group Press, Corte Madera, CA., 1996
3. William H. Murray and Chris H. Pappas, "Windows 95 and NT Programming with the Microsoft Foundation Class Library," AP Professional, Boston, 1996
4. Mark W. Spong and M. Vidyasagar, "Robot Dynamics and Control," John Willy & Sons, New York, 1989
5. K.S. Fu, R.C. Gonzalez, and C. S. G. Lee, "Robotics : Control, Sensing, Vision, and Intelligence," McGraw-Hill international Editions, New York, 1987.
6. A. Ailon and R. Segev, "Stable Observer-Based Trajectory Controller for Asymptotic Model Matching of a Rigid Robot," Journal of Optimization Theory and Applications, Vol.87 No 3, pp.517-538, December 1995.
7. Haruhiko Asada, and Kamal Youcef-Toumei , "Direct-Drive Robot",The MIT Press, Cambridge, 1987
8. Ilion A. Bonev, and Y.M. Yang, "Output Controller for Robot Trajectory Tracking System", Mechatronics Project, KJIST, June 14, 1997.
9. Craig C.R., and Akin D. L., "Cooperative Control of Two Arms in the Transport of an Inertia Load in Zero Gravity," IEEE Journal of Robotics and Automation, Vol 4, pp.414-419,1988.