

논문 2007-44IE-4-2

URC시스템과 연계한 지능형 로봇 시뮬레이터의 설계 및 구현에 관한 연구

(A Study On design & implementation of the intelligent robot simulator which is connected to an URC system)

남 상 엽*, 이 효 영**, 김 석 중**, 강 이 철***, 김 근 은***

(Sang-Yep Nam, Hyo-Young Lee, Suk-Joong Kim, Yi-Chul Kang, and Keun-Eun Kim)

요 약

URC의 개념은 “언제 어디서나, 나와 함께 하며, 나에게 필요한 서비스를 제공하는 로봇”으로 간단하게 정의할 수 있다. URC 기술은 다양한 로봇 단말이나 환경 내 컴퓨터와 URC 서버 간의 통신 프레임워크 기술이다. 본 논문에서는 지능형 로봇 시뮬레이션 툴을 이용하여 개발자가 쉽고 편리하게 지능적이고 능동적인 URC 서버와 연계하여 URC 로봇을 시뮬레이션 할 수 있는 시스템을 모델링하였다. URC 로봇 시뮬레이션 시스템은 다양한 로봇과 다양한 환경을 구성가능하며, 개발자가 다양한 개발 언어를 사용하여 제어 가능하도록 구성하였다. URC 시스템과 연계하여 다양한 로봇과 환경을 3D로 구성하고 시뮬레이션을 통해 다양한 콘텐츠 개발을 가능하도록 지능형 로봇 시뮬레이션 시스템을 설계 및 구현 하였다. URC 통신 프로토콜 및 URC 서버는 한국전자통신연구원(ETRI)에서 개발된 Planet v.1.2 ; Network Protocol , CAMUS(Context-Aware Middleware for URC Systems); URC Server, SAM(Service Agent Manager) v.1.2 ; Service API 모듈을 기반으로 하였다

Abstract

Concept of URC does “with me wherever when, and the robot” which provides necessary service to me can be simply defined. This paper uses URC technology, and various robots are implemented with a design. That is, we are going to implement that a user controls a virtual robot by communication between URC server with a design. We used an intelligent robot simulation tool, and a developer was easy, and it was intelligent, and we were connected to active URC server, and modeling did a system for simulation to be able to do an URC robot usefully. It was connected to an URC system, and various robots and environments were composed with 3D, and, in this paper, a design and implementation did an intelligent robot simulation system so that it was possible by various contents development through simulation. The URC communication protocol and the URC server were based on a Planet v.1.2 ; Network Protocol , CAMUS(Context-Aware Middleware for URC Systems); URC Server, SAM(Service Agent Manager) v.1.2 ; Service API module developed in Electronics & Telecommunications Research Institute (ETRI).

Keywords : URC, CAMUS, SAM, 지능형로봇, 로봇시뮬레이터

I. 서 론

현재 다양한 형태로 개발 중인 서비스 로봇이 우리

생활 속에 들어오기 위해서는, 사용자가 저가로 쉽게 로봇을 구입하여 다양한 서비스를 받을 수 있어야 한다. 이를 위해 로봇의 가격은 낮추는 반면 그 기능과 서비스는 다양화될 수 있도록 URC(Ubiquitous Robotic Companion)라고 하는 새로운 개념의 네트워크 기반 로봇에 대한 기대가 커지고 있다. 본 논문에서는 URC 시스템과 연계하여 다양한 로봇과 환경을 3D로 구성하고 시뮬레이션을 통해 다양한 콘텐츠 개발을 가능하도록 지능형 로봇 시뮬레이션 시스템을 설계 및 구현 하였

* 정회원, 국제대학
(KookJe College)

** 정회원, (주)ED
(ED Co. Ltd)

*** 정회원, 한국정보사회진흥원
(National Information Society Agency)

접수일자: 2007년10월24일, 수정완료일: 2007년12월1일

다. URC 통신 프로토콜 및 URC 서버는 한국전자통신연구원(ETRI)에서 개발된 Planet v.1.2; Network Protocol, CAMUS(Context-Aware Middleware for URC Systems); URC Server, SAM(Service Agent Manager) v.1.2 ; Service API 모듈을 기반으로 하였다 [1].

II. URC 시스템의 구조

URC 개념이 구현되기 위해서는 통신 네트워크와 고성능 로봇용 서버 등과 같은 컴퓨터 하드웨어 인프라가 구축되어야 한다. 또한 이러한 인프라 상에서 구동되는 소프트웨어 인프라가 필요하다. 이러한 인프라가 구성되면 사용자 환경에 맞는 전용 프로그램을 작성하여 로봇에 탑재해야 한다. 그러나 사용자가 원하는 요구사항이 다양하고, 상황에 적절한 URC 환경이 구축되기 위해서는 적절한 능력을 보유한 개발자가 다수 필요하게 된다^[2]. 개발자는 로봇이 활동할 상황을 가정하거나 직접 사용자를 방문하여 실제 환경에서 로봇을 동작시키기 위한 프로그램을 작성해야 한다. 이러한 문제점을 해결하기위해서 URC 환경을 구축하고 다양한 상황을 구성하여 URC 로봇을 미리 시뮬레이션 해보는 과정이 필요하다^[3].

따라서 URC 로봇을 시뮬레이션하기 위해서는 다양한 기능을 갖춘 가상의 URC 로봇과 가상의 환경이 필요하다. 그림 1은 본 논문에서 설계하여 구축하려는 URC 시스템 구성의 블록도이다^[4]. 사용자가 컨트롤러를 사용하여 로봇을 제어하면 URC서버를 통하여 로봇을 제어 할 수 있도록 구성하는 것이다.

CAMUS는 기존의 알고리즘 중심의 로봇에서 탈피하여 로봇이 공통으로 사용하는 부분을 서비스화(SA-Service agent)시켜서 제공하는 구조이다. ETRI의

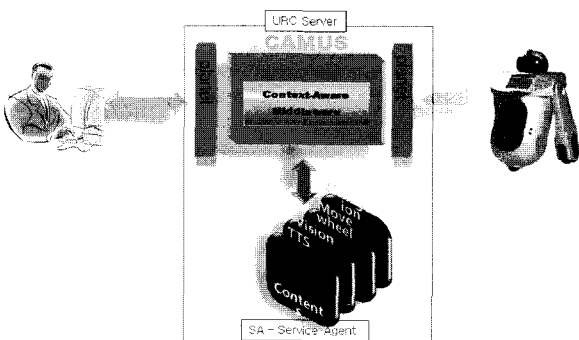


그림 1. URC 서버와 로봇의 구성
Fig. 1. URC server and configuration of a robot.

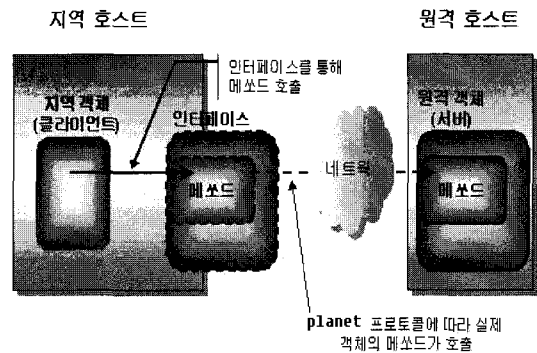


그림 2. CAMUS 시스템 구조
Fig. 2. CAMUS System Architecture.

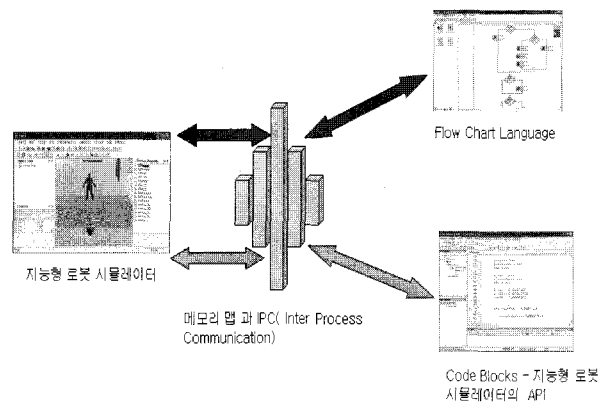


그림 3. 지능형 로봇 시뮬레이터의 메모리 맵과 IPC
Fig. 3. Memory map and IPC of intelligent robot simulator.

CAMUS의 기본 구조는 CORBA의 원격 객체 통신(분산형 객체)을 기반에 두고 있으며 Plant이라는 고유 통신 규약이 적용된다^[5]. 그림 2는 CAMUS 통신 프로토콜에 대한 구조이다. 지역 호스트의 지역객체인 로봇은 인터페이스를 통하여 원격 호스트의 원격객체(CAMUS)와 통신을 하며 또 다른 지역객체(Controller)는 원격 호스트의 원격객체에 접속을 하여 지역객체인 로봇의 제어권을 획득을 하여 원하는 동작을 지역 객체인 로봇에 보내면 원격 객체(CAMUS)는 지역객체인 로봇을 호출하여 원하는 구동을 할 수 있게 된다^[6].

그림 3처럼 로봇과 환경이 완성되면 플로우 차트나 C/C++언어와 API에 의하여 로봇 시뮬레이션이 이루어진다. 본 논문에서 구축하려는 지능형 로봇 시뮬레이터와 구동용 언어는 time slice가 Event time이며 IPC를 위하여 메모리 블록 넘김을 사용하여 프로세서간 통신을 한다. 그리고 본 논문에서 설계 구현할 지능형 로봇 시뮬레이터는 MS의 지능형 로봇 시뮬레이터인 MSRS와 연동 할 수 있도록 두 시뮬레이터 간의 IPC를 무리없이 위해 실제 물리 공간을 추상화한 가상공간과 가

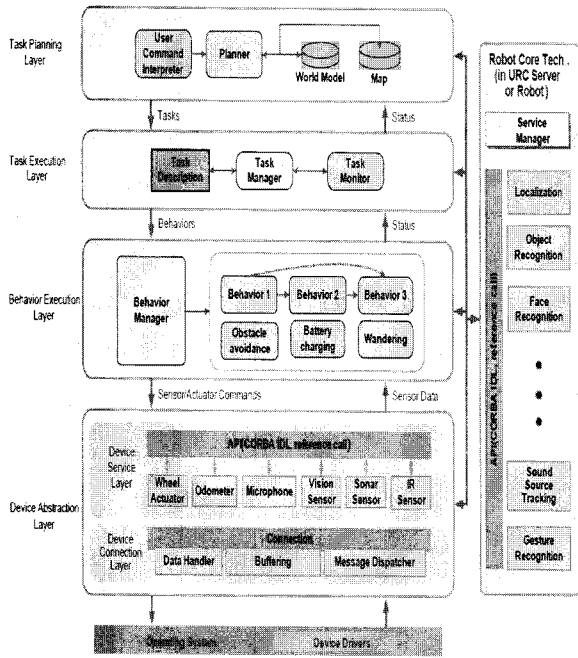


그림 4. URC 시스템 구조
Fig. 4. URC System Architecture.

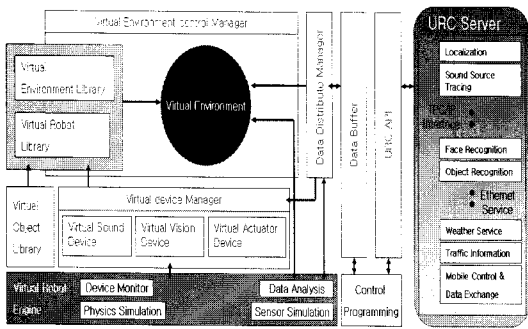


그림 5. URC 로봇 시뮬레이터 구조
Fig. 5. URC robot simulator Architecture.

상공간에서 동작하는 가상 로봇을 구성하고, 가상환경 (가상환경 = 가상공간 + 가상로봇)을 통해 발생하는 데이터에 의해 가상로봇이 동작하는 시뮬레이션 환경을 구성한다. 또한 URC 서버와 통신하기 위한 환경을 구성 가능하도록 별의 API 제공하여 개발자가 적절하게 프로그래밍 가능하도록 구성한다. 이를 위해 [그림 5] 같은 모듈을 구성 한다^[8].

- Dynamic Element Generator 모듈
- 3D 객체 상호간에 인식 기술 모듈
- 가상 디바이스 컴포넌트 모듈
- 가상공간 및 가상 로봇 구성 에디터 모듈
- 지능형 로봇 시뮬레이션 엔진 모듈
- URC API 모듈

이 통신을 하여 이기종간의 시뮬레이터의 연동을 하도록 구현한다^[7].

III. 지능형 로봇 시뮬레이션 모듈 설계

지능형 로봇 시뮬레이션 모듈 설계를 위해서 그림 2 처럼 로봇 시뮬레이션에 필요한 라이브러리를 생성을 위한 모듈을 설계하고 로봇 및 환경 구성 편집기를 설계하고 시뮬레이션 구성을 위한 통합환경을 구축한다.

이 과정이 끝나면 로봇 제어 프로그램들 C/C++ 및 플로차트언어를 이용하여 응용 프로그램을 작성하여 로봇 및 환경 시뮬레이션을 구현한다. 문제가 발생기 반복하여 수정한다.

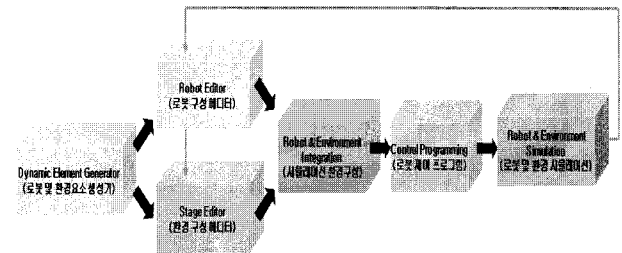


그림 6. 지능형 로봇 시뮬레이션 모듈 설계
Fig. 6. An intelligent robot simulation module design.

1.DEG(Dynamic Element Generator) 모듈설계
DEG는 3D 모델링 객체(로봇 설계 자료)를 이용하여 구성하고자 하는 객체의 특성에 따른 속성을 지정하고 물리적 동작형태를 설정할 수 있도록 설계하였다. DEG는 VEC(Virtual Environment Component)와 VRO(Virtual Robot Object)를 생성하는 기능을 가지고 있어야 한다. 따라서 생성되는 VEC와 VRO는 분류에 따라 라이브러리 형태로 저장되며, 사용자의 필요에 따라 생

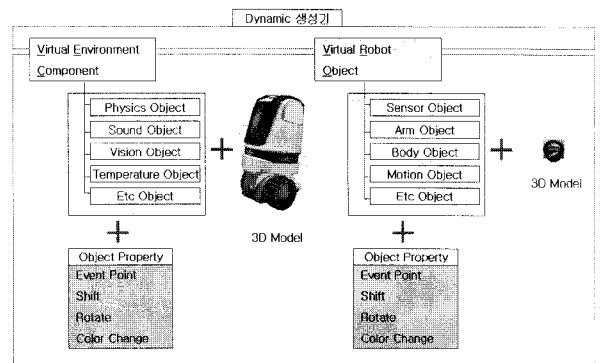


그림 7. Dynamic 생성기
Fig. 7. Dynamic generation tool.

성된 VEC와 VRO를 재결합하여 새로운 VEC와 VRO를 생성할 수 있도록 [그림 7]처럼 설계 하였다^[9].

2. 3D 객체 상호간에 인식 기술 모듈 설계

VEC는 역학적인 요소와 소리, 온도, 빛 등의 요소로 구성되며, VRO는 센서, 액추에이터 등의 요소로 구성된다.

이러한 객체간의 상호인식은 각 객체에서 출력되는 정보를 3D Graphics 환경에서 인식하고 가상환경을 구성하는 객체정보로 변경해야 시뮬레이션에 필요한 기본 정보로 사용가능 하다. 이를 위해 [그림 8]처럼 객체 상호간에 인식이 가능하도록 설계하고 구현하였다.

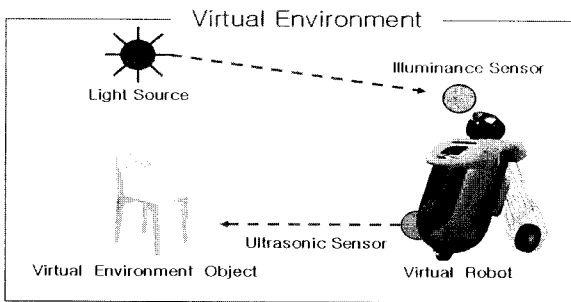


그림 8. 객체 상호 인식 기술
Fig. 8. Object mutual recognition technology.

3. 가상 디바이스 컴포넌트 모듈 설계

실제 로봇을 구성하는 하드웨어를 추상화하여 가상 디바이스 컴포넌트를 설계한다. 각 컴포넌트는 실제 디바이스와 동일하게 아날로그와 디지털 값을 입출력 가능하도록 구현하고 구성된 가상 컴포넌트는 VEC 또는 VRO에 맵핑되어 VEC와 VRO의 동작 형태에 영향을

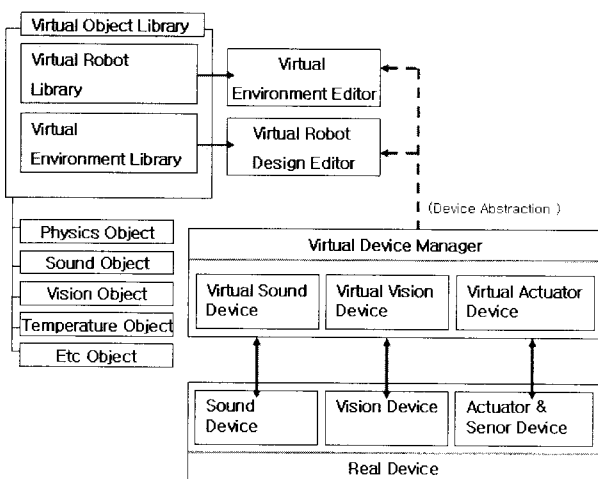


그림 9. 가상 미들웨어 컴포넌트 구조
Fig. 9. Virtual middleware component Architecture.

주며, URC 로봇의 특징을 갖는 구성을 가지도록 [그림 9]처럼 설계 하였다^[10].

4. 가상공간 및 가상로봇 구성 에디터 모듈 설계

그림10처럼 가상공간 및 가상로봇 구성 에디터는 사용자가 2D환경에서 가상환경을 편리하게 구성하고 이를 3D 환경에서 시뮬레이션 가능하도록 설계 하였다.

2D로 구성된 라이브러리에서 사용자가 가상환경에 삽입하고자 하는 요소를 선택하고, VEE(Virtual Environment Editor)와 VRDE(Virtual Robot Design Editor)에 삽입한다. VEE와 VRDE에서는 삽입된 데이터를 VEC와 VRO의 라이브러리에서 찾아내고, 해당 데이터를 3D로 구성된 가상환경의 특정 위치에 2D에디터의 위치에서 유추하여 위치시키도록 [그림 10] 처럼 설계 하였다^[11].

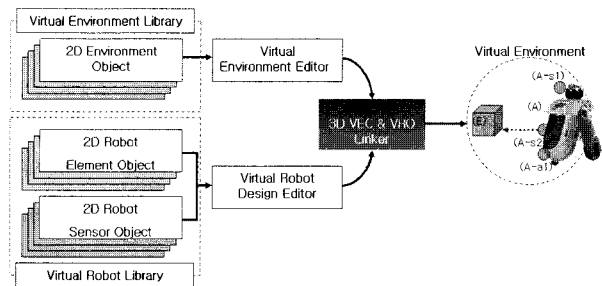


그림 10. 가상환경 및 가상로봇구성 에디터 구조
Fig. 10. Virtual environment and virtual robot configuration editor Architecture.

5. 지능형 로봇 시뮬레이션 엔진 모듈 설계

그림 11처럼 지능형 로봇 시뮬레이션 엔진은 가상환경에서 발생하는 데이터와 메시지를 VEC와 VRO의 객

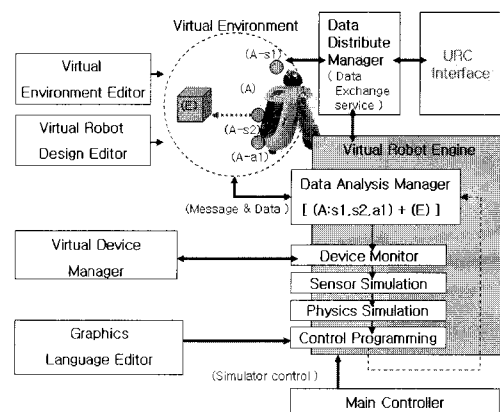


그림 11. 지능형로봇 시뮬레이션엔진 구조
Fig. 11. Intelligent robot simulation engine Architecture.

체로 구분하고, 가상 미들웨어 컴포넌트, 제어명령 및 URC 서버에서 제공된 데이터와 조합하여 가상 로봇의 다음 행동을 결정하도록 설계하였다.

가상 로봇의 동작은 가상환경에 영향을 주고 변화된 가상환경의 데이터와 메시지는 로봇 시뮬레이션 엔진으로 feedback 되어 가상로봇의 다음 행동을 결정하도록 feedback cycle 구조를 갖는다^[12].

6. URC API 모듈 설계

그림 12처럼 URC를 이용하여 지능로봇 시뮬레이터의 가상 디바이스 및 서비스와 연동 가능하도록 URC 프로토콜을 기준으로 API 모듈을 설계 하였다. URC API모듈은 가상의 로봇을 제어하기위한 가상 디바이스와 SAM은 Wrapping하여 구현된다.

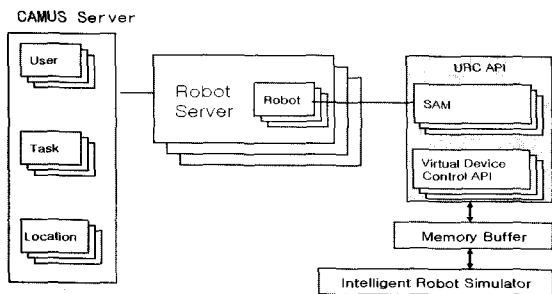


그림 12. URC API 모듈
Fig. 12. URC API Module.

IV. 구현 결과 및 검토

URC 로봇의 시뮬레이션하기 위해 설계하여 구현한 시뮬레이터를 이용하여 다양한 형태로 로봇을 제어하기 위해 지능형 로봇 제어용 컴파일러 모듈 구현(C, C++, Java, Flowchart Language), VEC 및 VRO 라이브러리 모듈 구현, URC 콘텐츠 연동 모듈 구현 모듈을 구현하였다.

1. 로봇 제어용 시뮬레이터 전체 프로그램 구현

그림 13처럼 메인 프로그램에서 시뮬레이션을 시작 하면 컴파일러 프로그램에서 시작을 알리는 메시지를 보내면 사용자가 작성한 프로그램을 실행하여 데이터 테이블에 접근하게 된다. 데이터 테이블을 통하여 시뮬레이션동작을 한다. 그리고 컴파일러 프로그램에서 디버깅을 시작하면 디버깅 상태를 계속해서 메인 프로그램으로 전송한다.

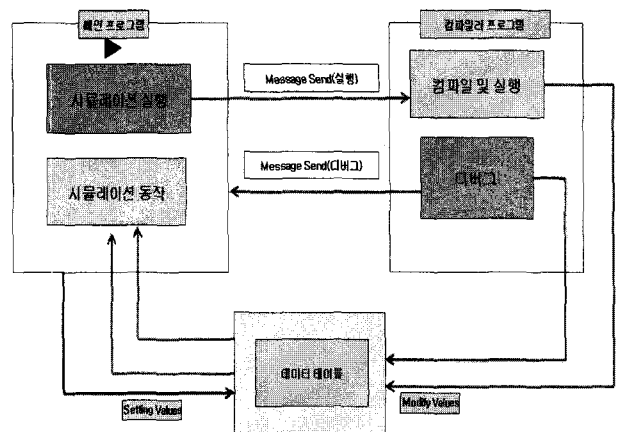


그림 13. 로봇 제어용 시뮬레이터의 동작
Fig. 13. An action flow chart of robot control simulator.

컴파일러 프로그램에서 받은 메시지를 보고(Break Point) 시뮬레이션을 정지 다시 실행을 반복한다.

메인 프로그램은 로봇 센서, 모터의 속성값 및 환경 센서 속성 값 등을 세팅해 준다.

2. 지능로봇 제어용 컴파일러 모듈 구현

가상로봇을 C, C++, Java, Python, Flowchart Language를 이용하여 제어할 수 있도록 구현하였다.

로봇 제어용 컴파일러는 로봇의 특정 디바이스를 제어

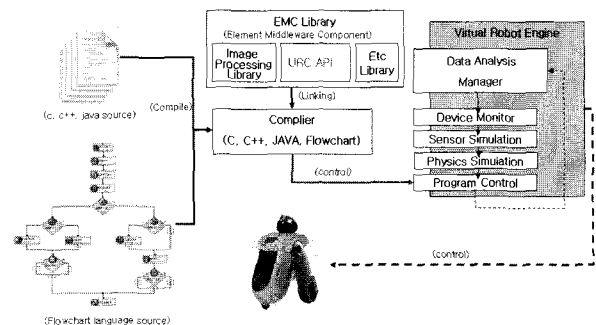


그림 14. 로봇 제어용 컴파일러 동작구조
Fig. 14. Compiler action structure of robot control use.

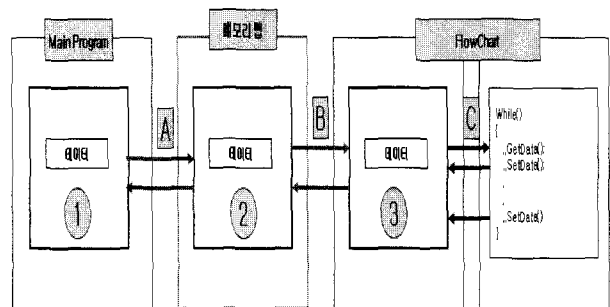


그림 15. 데이터 처리 구조
Fig. 15. Data process structure.

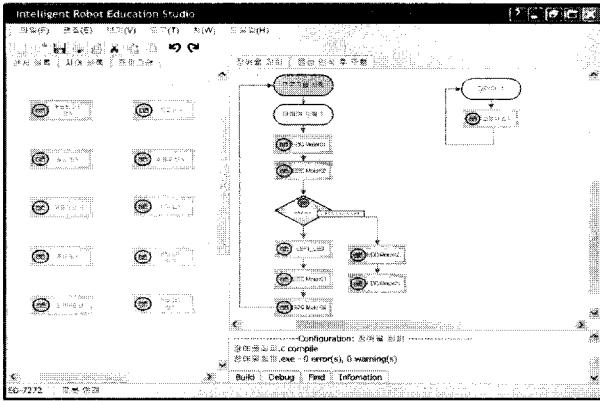


그림 16. Flowchart 형태의 Graphic Language
Fig. 16. Graphic Language of a Flowchart form.

하기 위해 가상 요소별 디바이스 컴포넌트를 함수로 호출하여 사용 가능하도록 [그림 14]처럼 구현하였다.

데이터 처리과정은 [그림15]처럼 ①의 데이터를 ②에 복사하고 플로우차트에 메시지를 전송한다. 메시지를 받은 플로우차트에서 ②의 데이터를 ③에 복사한다. 플로우차트의 한 사이클동안 ③의 데이터를 접근하여 수정한다. 한 사이클이 끝나면 ③의 데이터를 ②에 복사하고 메인에 메시지를 전송한다. 메시지를 받은 메인에서 ②의 데이터를 ①에 복사한다. 데이터 처리과정은 이 과정을 반복한다^[12].

Flowchart Language는 비숙련 사용자의 제어 프로그램 작성을 위해서 Graphics Language 형태로 [그림 16]처럼 설계하여 구현하였다.

3. VEC 및 VRO 라이브러리 모듈 구현

VEC 및 VRO Editor를 사용하여 가상환경을 구성하기 위한 다양한 형태의 객체를 라이브러리 형태로 [그림 17]처럼 구현하였다.

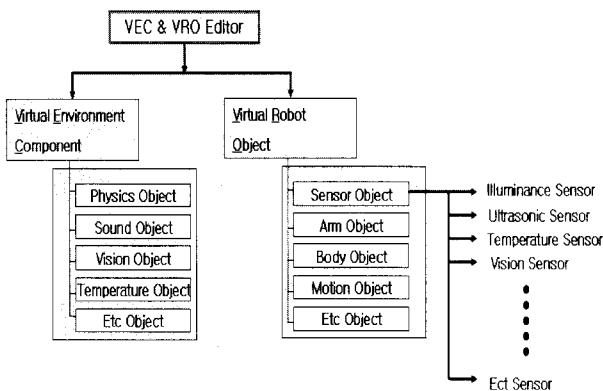


그림 17. VEC 및 VRO 라이브러리 구조
Fig. 17. VEC and VRO library structure.

4. URC 콘텐츠 연동 모듈 구현

그림 18처럼 사용자가 다양한 URC 환경과 유연하게 연동 가능 하도록 시뮬레이터 및 로봇 플랫폼에 URC 콘텐츠 처리 API를 연결하여 다양한 URC 콘텐츠 실행 가능하도록 구현하였다.

그림 19는 지능형 로봇 시뮬레이터의 구현 사례이다. 로봇의 센서와 제어기를 PC에서 시뮬레이션 하면서 URC를 이용하여 제어 소프트웨어를 개발할 수 있다. 시뮬레이션은 로봇 하드웨어로는 불가능한 다양한 테스트를 할 수 있어 강인한 소프트웨어를 개발할 수 있다. 시뮬레이터를 이용하여 제어 소프트웨어 개발이 어느 정도 완료되었다면, 시뮬레이터를 로봇 하드웨어와 디바이스 드라이버로 교체할 수 있다. 이는 제어 소프트웨어에서 연결 정보를 수정하면 된다. 로봇 하드웨어와 연결된 제어 소프트웨어는 시뮬레이터에서 예측하지 못한 문제들을 가지고 있으면 제어 소프트웨어는 URC에서 개발되기 때문에 로봇에 탑재하여 개발하는 것보다

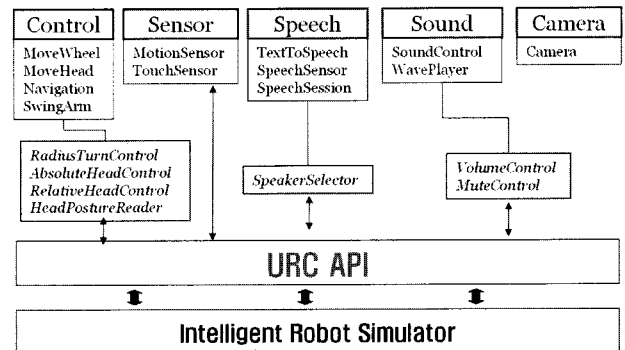


그림 18. URC 콘텐츠 연동 구조
Fig. 18. URC contents working together Architecture.

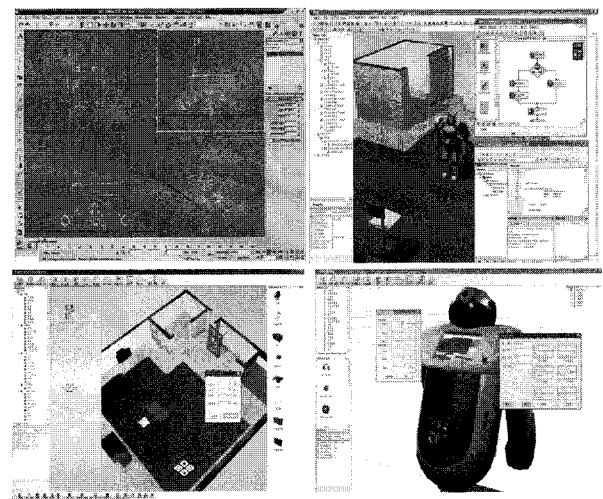


그림 19. 지능형 로봇 시뮬레이터의 구현 예제
Fig. 19. An implementation example of intelligent robot Simulator.

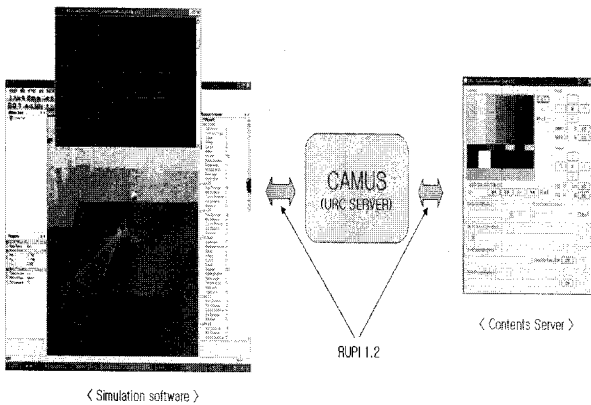


그림 20. 로봇 시뮬레이터와 URC 서버 연동
 Fig. 20. Interaction of Simulator and Urc server.

디버깅이 편리하다. 하드웨어 테스트가 완료된 경우, URC에서 개발한 제어 소프트웨어를 로봇으로 탑재하여, 로봇이 독립적으로 기능을 수행할 수 있다.

지능형 로봇에서 사용 중인 센서와 액추에이터를 같은 클래스에 속하는 다른 종류로 바꾸어 제어 소프트웨어가 정상적으로 동작하는지 확인할 수 있다. 시뮬레이터는 Sensor Device와 Vehicle Device를 시뮬레이션 할 수 있고 profile의 교체만으로 특성이 다른 다양한 센서들을 만들어 낼 수 있다.

그림 20은 지능형 로봇 시뮬레이터와 URC Content 서버를 CAMUS 통해 중개하여 구현된 실제 시뮬레이션의 사례이다.

V. 결 론

지능형 로봇 시뮬레이터와 3D엔진기술을 통합하여 3D 가상환경에서 지능형 로봇의 요소기술 및 제어 기술에 대한 개발 및 연구를 용이하게 수행할 수 있는 시뮬레이션 소프트웨어를 구현하였다. 또한 URC Architecture를 적용하여 URC의 다양한 상황을 시뮬레이션 가능하도록 API 인터페이스를 구현 하였다. 본 연구를 통해 URC를 활용한 지능형 로봇 시스템의 설계 및 개발에서 좀 더 효율적이고 용이한 구현과 시뮬레이션을 통한 다양한 URC 서비스 모델이 개발이 가능할 것으로 기대된다.

참 고 문 헌

[1] Aekyung Moon, Minyoung Kim, Hyoungsun Kim, Kang-Woo Lee , Hyun Kim, "Development

of CAMUS based Context-Awareness for Pervasive Home Environments", International Journal of Smart Home Vol. 1, No. 1, January, 2007.

[2] Brooks, R.A., "A robust layered control system for a mobile robot," IEEE Journal on Robotics and Automation, Vol.2, No.1, 1986.

[3] 남상엽, 이효영, 김석중, "지능형 로봇 설계 및 개발을 위한 로봇 시뮬레이터의 구현에 관한 연구", 대한전자공학회 합동학술대회, pp113-116, May, 2007.

[4] H. Kim, Y. Cho, and S. Oh, "CAMUS: A Middleware Supporting Context-Aware Services for Network-Based Robots," IEEE Workshop on Advanced Robotics and Social Impacts, 2005.

[5] K. Min, K. Nam, and J. Kim, "Multilevel Location Trigger in Distributed Mobile Environments for Location-Based Services," ETRI Journal, vol.29, no.1, 2007, pp.107-109.

[6] Brooks, R. A., "Intelligence without representation," Artificial Intelligence, Vol.47, pp.139-160, 1991.

[7] A. Ranganathan and R.H. Campbell, "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments," LNCS (2672), 2003, pp. 143-161.

[8] Aekyung Moon, Hyoungsun Kim, Hyun Kim, Soowon Lee , "Context-Aware Active Services in Ubiquitous Computing Environments", ETRI Journal, Volume 29, Number 2, April 2007.

[9] Jinwook Huh, Woong Sik Chung, Sang Yep Nam, and Wan Kyun Chung, "Mobile Robot Exploration in Indoor Environment Using Topological Structure with Invisible Barcodes", ETRI Journal, Vol.29, No.2 ,pp189-200, April 2007.

[10] C.S. Hong, J.M. Cho, K.W. Lee, Y.H. Seo, H.S. Kim, H. Kim, "Context Management Technology for Ubiquitous Robotic Companion", ETRI, Vol. 22, Apr, 2007.

[11] Sang Yep Nam, Keun-Eun Kim, Yi-Chui Kang, Hyo-Young Lee, Suk-Joong Kim, "Study on the URC robot simulation that used intelligent robot simulator", URAI 2007, pp644-647, 22 Nov, 2007.

[12] 남상엽, 이효영, 김석중, 강이철, 김근은, "지능형 로봇 시뮬레이터를 이용한 URC 로봇 시뮬레이션에 관한 연구", 2007년도 대한전자공학회 추계종합학술대회 논문집II, 제31권 제2호 p858-859, 24. Non. 2007.

— 저 자 소 개 —



남 상 엽(정회원)

2002년 단국대학교 전자공학과
박사 졸업.

1987년~1992년 삼성종합기술원
정보시스템연구소
주임연구원

1992년~1998년 (주)모토로라
반도체통신 기술연구소 차장

1998년~현재 국제대학 정보통신과 교수
<주관심분야 : 지능형로봇, 임베디드 시스템,
USN/RFID, 홈네트워크>



이 효 영(정회원)

2000년 전주대학교 산업공학과
학사 졸업.

2002년 전남대학교 산업공학과
석사 졸업.

2002년~현재 (주)ED 기술연구소
소프트웨어팀 팀장

<주관심분야 : 지능형로봇, 시뮬레이션SW, 유비
쿼터스 홈 네트워크>



김 석 중(정회원)

1992년 국민대학교 전자공학과
학사 졸업.

1994년 국민대학교 전자공학과
석사 졸업.

1994년~2001년 터보테크
CNC개발

2001년~현재 (주)ED 기술연구소 본부장/이사
2004년~현재 지능형로봇개발 총괄 병행
<주관심분야 : 로봇틱스, USN/RFID>



강 이 철(정회원)

1999년 동국대학교 조경학과
학사 졸업.

2001년 부산대학교 멀티미디어
학과 공학석사 졸업.

2007년 부산대학교 멀티미디어
학과 박사과정 수료.

2007년 현재 한국정보사회진흥원 선임연구원
<주관심분야 : HRI, 인공지능, 네트워크 로봇>



김 근 은(정회원)

1990년 광운대학교 전자공학과
학사 졸업.

1997년 콜로라도대학교
전자공학과 석사 졸업.

2007년 부산대학교 컴퓨터공학과
박사 과정

2007년 현재 한국정보사회진흥원 책임연구원
<주관심분야 : HRI, 인공지능, 네트워크 로봇>