

다채널 디스플레이에서 X3D와 모션체어를 이용한 수중운동체 시뮬레이터

허필원*, 양정삼**, 한순흥***

An Underwater Simulator Using X3D and a Motion Chair in a Multi-channel Display Room

Pilwon Hur*, Jeongsam Yang** and Soonhung Han***

ABSTRACT

A submarine good military weapon because of its confidentiality and intimidating power. Therefore, training warfighters how to maneuver submarine is very important. Because submarine is very expensive and has regional and temporal limitations, M&S(Modeling and Simulation) can be a good alternative. However, as the existing M&S systems of submarine generally use expensive commercial software and dedicated hardware, which cause the warfighters to take troubles to visit the secured places, and then to train themselves during limited time slots. Also, many M&S systems have only one-channel display system which reduces the sense of immersiveness. Another problem is that many heterogeneous simulators can hardly be used as an integrated system. To solve these problems, X3D, a platform-independent and open standard graphic file format, is used with the general-purpose PCs. To increase immersiveness, multi-channel display system and a motion chair are used. Finally, HLA/RTI is used to integrate individual components of the simulator. All of these are verified through experiments.

Key words : Modeling and simulation, Motion chair, Multi-channel, underwater vehicle, X3D

1. 서 론

잠수함과 잠수함에 탑재된 무기체계는 수상 및 수중 표적을 겨냥한 기존의 무기체계 개념에서 벗어나, 대지 육상 공격 능력을 갖는 수중발사 전략 유도무기가 잠수함에 채택됨에 따라서, 독립된 무기체계로서의 역할이 강조되어 왔다. 우수한 무기체계를 탑재한 잠수함과 더불어, 잠수함을 효과적으로 운용하는 것이 전쟁에서의 승패에 영향을 미친다. 그러므로 장병들에게 잠수함을 잘 운용하는 방법을 교육시키는 훈련이 중요하다. 장병들이 실물 잠수함을 직접 타고 훈련을 하는 방법은 실제 상황과 같은 소리와 진동을 느끼면서 직접 장비를 조작할 수 있기 때문에, 훈련의

효과가 크다. 하지만, 잠수함이라는 장비가 고가이고, 바다라는 지역적 한계, 그리고 실제 훈련을 나갈 경우 최소 몇 개월이 소요되는 시간적 한계 때문에 현실적으로 어려움이 많다. 이에 대한 대안으로 잠수함의 훈련 환경을 가상현실 속에 비슷하게 꾸미고, 잠수함의 움직임을 모사하는 M&S(Modeling and Simulation) 방법이 있다.

해군에서 새로 개발한 잠수함이 수중에서 얼마나 빠른 속도로 움직일 수 있는지를 알고 싶을 경우, 실제의 잠수함을 물 속에 집어 넣고 최대의 마력으로 추진시켜 이때의 잠수함의 속도를 측정하면 된다. 이런 경우에는 M&S가 크게 필요하지 않을 것이다. 그러나 잠수함을 향하여 어뢰가 발사되었을 경우, 어느 정도의 확률로 어뢰를 회피하여 생존할 수 있는가를 알고 싶은 경우에는, 잠수함을 실제로 타고 실험을 할 수 없다. 이와 같은 상황에서는 M&S 방법을 이용하여 가상의 시뮬레이션을 수행하는 방법을 적용할 수 있다. 그러나 시뮬레이션 시스템은 장병들의

*University of Illinois at Urbana-Champaign

**교신저자, 종신회원, 아주대학교 산업정보시스템공학부

***종신회원, KAIST 기계공학과

- 논문투고일: 2007. 07. 24

- 심사완료일: 2007. 10. 08

잠수함 전술 운용 훈련에 실질적으로 별로 도움이 되지 못하는 경우가 많다^[1]. 그 이유로는 기존의 잠수함 M&S는 가시화를 위해서 고가의 상업용 툴을 사용하며, 이를 활용하기 위해서는 전용 하드웨어가 필요하기 때문이다. 또한 이러한 시스템을 관리하기 위해서 특별한 관리 장소가 필요하며, 유지보수에도 많은 비용이 들어간다. 정해진 장소와 정해진 시간에만 사용해야 하므로 이동의 불편함과 활용 시간상의 제약도 따르게 된다.

본 논문에서는 (1) 고가의 전용시스템에 의존적인 기존의 잠수함 M&S 가시화 대신에, 일반적으로 쉽게 사용할 수 있는 저가이며 플랫폼 독립적인 가시화 시스템을 구성하고, (2) 단일 채널의 가시화 방법 대신에 다채널의 가시화와 함께, 모션제어를 적용하여 몰입감을 높이고, (3) 서로 다른 환경의 시뮬레이터들과의 연동이 가능한 프레임워크를 적용함으로써, 시뮬레이터들의 호환성과 재사용성을 증대시킬 수 있는 통합 시스템을 제안한다.

본 논문의 구성은 다음과 같다. 2절에서는 과거에 수행된, 가상환경에 대한 가시화 방법과, 군사 훈련 목적으로 진행된 M&S 연구에 대한 소개를 한다. 3절에서는 시뮬레이터를 구성하는 컴포넌트들을 통합하기 위해서, 공통의 프레임워크인 HLA(High level architecture)를 기반으로, 다채널 디스플레이와 모션제어의 설계 방법을 제안한다. 4절에서는 iCAVE 환경에서 작동하는 수중 운동체를 위한 시뮬레이터를 소개하고, 이 시뮬레이터를 이용하여 실험한 결과를 보여준다.

2. 관련 연구

2.1 가상환경을 구축하기 위한 가시화 방법

3차원 가상환경을 가시화하기 위해서 다양한 그래픽 라이브러리를 이용하고 있고, 모델링 된 가상환경을 저장하기 위해서 다양한 그래픽 파일 포맷이 이용되고 있다. OpenGL과 Direct3D와 같은 저수준(Low level)의 그래픽 라이브러리는, DDI(Device drive interface)를 거쳐 그래픽 하드웨어에 가깝게 접근하기 때문에, 렌더링의 질을 높은 반면에, 저 수준의 그래픽 함수 호출에 따른 복잡한 구현이나 최적화 과정이 필요하다. Java를 기반으로 한 저수준 그래픽 라이브러리인 JOGL(Java bindings for OpenGL API)은 C 언어 기반의 절차식 그래픽 언어인 OpenGL을 Java에서도 완벽하게 사용할 수 있도록, OpenGL을 일대일 매핑에 의해 번역을 한 것이다. 그러므로 JOGL 또한

OpenGL과 Direct3D와 같은 저수준 그래픽 라이브러리의 특징을 지닌다. 한편, Vega, Java3D, OSG(Open Scene Graph) 등과 같은 고수준(High level)의 그래픽 라이브러리들이 가상환경에 대한 가시화에 사용된다.

Vega는 실시간 시뮬레이션을 목적으로 OpenGL 기반의 렌더링 툴킷인 Performer의 기능을 향상시킨 상업용 가시화 개발 툴킷이다. Christianson는 Vega와 Java3D를 이용하여 동일한 모델을 가시화 하면서 두 개의 툴의 성능을 비교하였다^[2]. 그 결과 Vega는 비록 플랫폼에 종속적이고 고가이지만, 연속된 장면(Scene)에 대한 프레임율(Frame rate)과 같은 성능 면에서 Java3D에 비해 좋은 결과를 보였다. 반면 Java3D는 플랫폼에 독립적이고 무료로 사용할 수 있지만, 성능은 Vega 보다 떨어짐을 보였다.

Java3D는 웹 기반의 가시화를 지원하기 위해 개발된 그래픽 라이브러리로서, OpenGL의 주요 기능을 Java 기반의 툴킷으로 개발되었기 때문에, 플랫폼 독립적이며 저가형 가시화에 적절한 가시화 방법이다. 저수준 그래픽 라이브러리인 OpenGL의 상위 계층에 얹혀져 있는 Java3D는, 사용자가 불필요한 그래픽 렌더링에 관심을 갖기 보다는 고수준의 장면그래프(Scene graph)의 설계 및 운용에만 집중을 하게 함으로써, 코딩의 양을 줄여 개발기간을 단축한다. 장면그래프란 장면의 정보를 구성하는 사물, 그 사물의 속성, 그리고 그 사물 주위에 존재하는 다른 사물들 간의 관계 등을 계층구조 형식으로 나타낸 자료구조를 말하며, 사물의 추상화나 조작을 편리하게 한다. Salisbury는 웹 상에서 시뮬레이션 가시화를 위해 Java3D를 이용하였다^[3]. 그러나 OpenGL에서는 가능하지만 Java3D에서는 사용할 수 없는 기능들이 일부 존재하며, 이로 인해 장면이나 렌더링을 제어하지 못하는 경우가 발생된다. 또한 렌더링 자체는 OpenGL과 같은 저수준 그래픽 라이브러리를 사용하지만, 프로그램 로직 등 나머지 모든 기능들은 Java를 사용하게 되므로 성능과 속도가 떨어진다. 특히 Java 런타임, Java3D 런타임 그리고 응용프로그램의 빌딩(building) 과정에서 모두 객체를 생성하게 되는데, 이런 객체들은 더 이상 사용되지 않을 경우에 Garbage 형태가 된다. 이 경우 Java 가상 머신(Java virtual machine)은 Garbage collection을 수행하여 Garbage들을 모은다. 따라서 Garbage collection이 수행되는 동안에는 애니메이션이 중간에 멈추므로 장면의 사실성이 떨어진다.

OSG는 OpenGL 기반의 3D 그래픽스 개발 툴킷으로서 객체지향 방식에 따라 개발되었다. OSG는 Vega

와 Performer 보다 이해하기 쉽고, 개발에 용의하며 표준 C++ 기반으로 작성되었기 때문에, 여러 플랫폼에서 재사용이 가능하지만, 웹 기반의 가시화가 어렵다는 단점이 있다.

위에서 소개한 그래픽스 라이브러리를 이용한 방법과 더불어, 그래픽 포맷을 가시화에 이용한 사례도 있다. 동적인 장면그래프를 지원하는 VRML97은 국제 표준인 3D 가시화 포맷이다. Li 등은 외과 수술 트레이닝을 위하여 웹 기반의 가상현실 시뮬레이터를 개발하였다^[4]. 이 때 가시화의 포맷으로 VRML을 사용하였고, 동적인 시뮬레이션을 위해서 EAI(External authoring interface)를 사용하였다. Roberts는 동일한 화면을 다른 시각으로 동시에 볼 수 있는 다채널의 가시화 방법을 제시하였다^[5]. 그러나 VRML은 장면그래프를 구성하는 장면 노드들을 개별적으로 관리하지 못하고, 불필요한 노드들까지 모두 가지고 있기 때문에, VRML 파일의 크기가 큰 것이 단점이다. 그리고 상호 운용성에 대한 문제점과 의미정보를 체계적으로 구조화 하기 어려운 단점이 있다.

또 다른 국제표준인 SEDRIS는 정적인 데이터를 저장하는 데이터베이스를 갖추고 있기 때문에, 환경 데이터에 대한 의미정보와 관계정보를 명확하게 표현하여 저장할 수 있어서, 서로 다른 데이터 포맷 간의 교환 수단으로 사용된다. 문홍일은 SEDRIS를 이용하여 디지털 생산을 위한 시뮬레이션 환경을 구축하였다^[6]. 디지털 합성된 환경 데이터와 각 사물 간의 의미정보와 관계정보를 저장하기 위해서 SEDRIS는 좋은 방법이 될 수 있지만, SEDRIS에 저장되어 있는 3D 데이터를 가시화 하기에는, 사용 가능한 SEDRIS 뷰어들의 기능이 빈약한 것이 단점이다.

Web3D 컨소시엄에서 제안한 VRML의 확장 버전인 X3D(Extensible 3D)는 XML 기술을 사용하여 VRML 데이터를 표현할 수 있다^[8,21,22]. 따라서 VRML97과 호환이 가능하고, XML을 기반으로 하였기 때문에, 의미정보에 대한 체계적 구조화가 가능하다. 또한 MPEG4와 같은 표준들과의 상호 운용성을 제공한다. 고급 렌더링에 필요한 노드의 추가로 가시화의 질도 높아졌으며, 텍스트 형식뿐만 아니라 바이너리 형식의 지원으로 파일의 용량이 작아지고 파일 압축이 가능하여, 웹 상에서의 전송과 실행속도가 빨라졌다^[7]. 이러한 장점으로 인해 최근 X3D를 가시화에 이용한 연구들이 진행되고 있다^[8-11].

본 연구에서는 가시화에 필요한 그래픽 파일 포맷으로 X3D를 사용하며, 브라우저로는 OpenGL이나 JOGL의 상위 계층에서 장면그래프 사용을 가능하게

하는 Java3D로 구현된 Xj3D를 사용함으로써, 저가의 플랫폼 독립적인 시스템을 구현한다.

2.2 군사 훈련 목적의 M&S 연구

미 국방성은 소그룹 단위의 군사 훈련을 목적으로, 네트워크로 연결된 저가의 시뮬레이터를 만들기 위해 SIMNET(Simulator networking)을 제시하였다^[12]. 분산 네트워크 형태의 SIMNET은 병사들에게 시뮬레이션 훈련을 통해, 위험한 전쟁 시나리오를 체험할 수 있는 가상 환경을 제공한다. SIMNET 기반의 DIS(Distributed interactive simulation)는, SIMNET과 마찬가지로 저가의 비용으로 분산 시뮬레이션이 가능하지만, SIMNET 보다 많은 수의 시뮬레이션이 분산 네트워크로 연결된 가상 환경에 참여할 수 있다^[13]. 그러나 DIS가 가지고 있는 패킷(Packet) 정의 집합이 넓기는 하지만, 모든 종류의 시뮬레이터들에게 적용될 만큼 보편적이지는 못하다.

미 국방성 산하 기관인 DMSO(Defense modeling and simulation office)는 DIS의 단점을 개선하고, 서로 다른 목적을 위하여 개발된 다수의 시뮬레이터들 간의 상호운용성(Interoperability)과, 시뮬레이션 컴포넌트의 재사용(Reuse)이 가능하도록 HLA(High level architecture)를 제안하였다^[14,15]. HLA는 분산 시뮬레이션 개발의 효율성 향상을 위한, 객체 기반의 분산 시뮬레이션 소프트웨어 구조이며, IEEE 1516 표준이다. HLA에서는 개별 시뮬레이션 시스템이 상호 연동되고, 시스템 구성 요소들이 재사용될 수 있도록 개념적인 정의를 내리고 있다^[16]. HLA는 광범위한 분산 환경에서 다양한 시뮬레이터들이 상호간에 호환성을 가지고 긴밀한 연동 관계를 통해 동작한다. 또한 HLA 기반으로 개발된 시뮬레이터들은 향후 추가로 개발된 시뮬레이터와 연동하기 쉽고, 기존의 컴포넌트 모듈을 재사용하기 용이한 프레임워크이다. 미국 방성은 2001년 이후 개발된 모든 국방분야 시뮬레이션들이 HLA를 따르도록 방침을 정하고 있으며, 현재 SIMNET 또는 DIS를 따르는 응용프로그램들을 HLA의 구조로 변환하는 연구가 진행되고 있다^[16]. 본 논문에서도 분산환경에 적용하는 시뮬레이터를 개발하기 위하여 HLA를 사용한다.

3. 다채널 디스플레이와 모션제어의 설계

3.1 시스템의 개요

수중운동체를 위한 시뮬레이터를 개발하기 위하

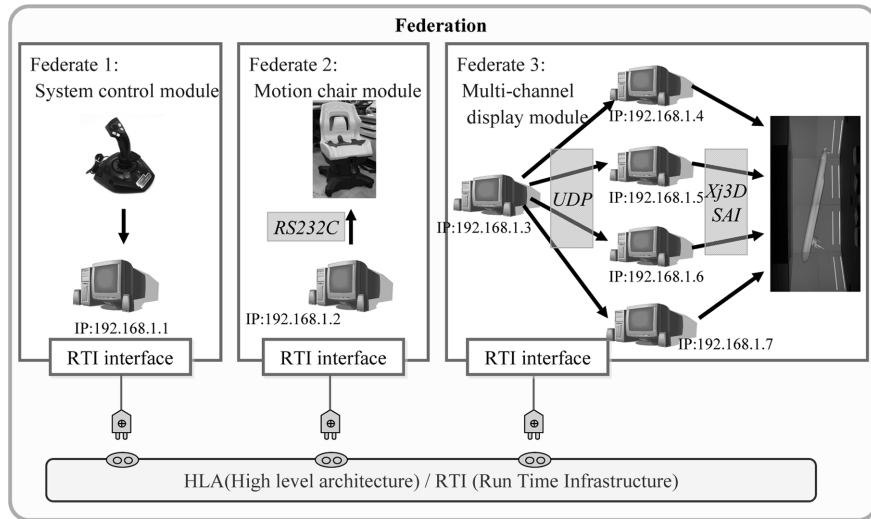


Fig. 1. Structure of integrated simulation system.

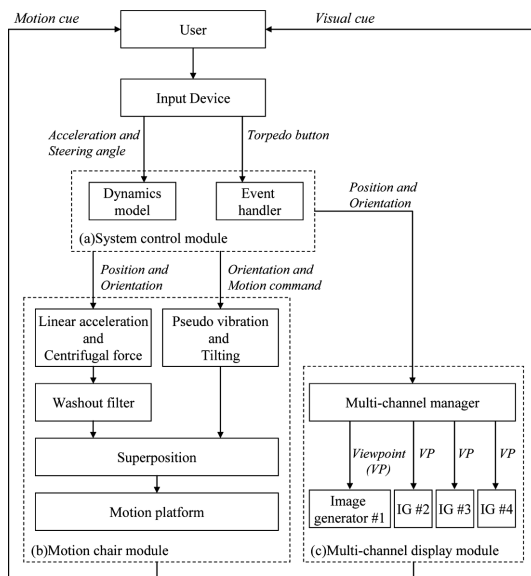


Fig. 2. Work process of integrated simulator.

여, 분산 환경에서 존재하는 시뮬레이션 컴포넌트들 상호간에 실시간 연동을 위한 HLA 프레임워크 개념을 도입하였다. 시뮬레이터는 Fig. 1에서 보는 바와 같이, 3개의 Federate들(시스템 제어 모듈, 모션제어 모듈, 다채널 디스플레이 모듈)로 구성되고, 이들 Federate들은 미들웨어이며, IEEE1516 표준인 RTI(Run-time infrastructure)를 통해 실시간으로 연동된다. 3개의 Federate와 RTI로 구성된 통합 시뮬레이션 시스템인 Federation이 구동되면, RTI는 Federate 상호간에 동기화에 필요한 메시지 교환을 제어한다. 각

각의 Federate는 다른 Federate들과 정보를 주고 받을 수 있도록, 자신의 시뮬레이션 시간을 관리한다. 본 논문에서는 통합 시뮬레이션 시스템을 구성하기 위하여, 7개의 독립된 IP 주소를 갖는 PC들을 클러스터링(Clustering)하였다. Fig. 2는 아군의 잠수함이 적군 잠수함의 위치를 파악하여 어뢰의 사정 거리까지 접근한 후, 어뢰를 발사하는 시나리오를 적용하기 위한 시뮬레이터의 구동 프로세스를 보여준다.

3.2 분산 시뮬레이션을 위한 Federation의 구성

3.2.1 시스템 제어 모듈

시스템 제어 모듈은 모션제어 모듈과 가시화 모듈을 비롯하여 전체 시스템에 대한 통제를 하는 역할을 한다. 사용자는 입력장치인 조이스틱을 이용하여 잠수함의 위치를 실시간으로 조작할 수 있다. 예를 들어 잠수함을 특정 수심까지 잠향시키기 원할 경우에, 사용자는 조이스틱을 조작하여 잠수함이 특정 수심까지 움직이도록 명령을 내린다. 그리고 이 명령을 전달받은 시스템 제어 모듈은, 잠수함의 현재 위치와 자세를 새로운 위치와 자세로 수정한 후에, 모션제어 모듈과 가시화 모듈에 전달한다.

사용자는 Fig. 2에서 보는 바와 같이 모션제어에서 얻은 Motion cue와 가시화 모듈에서 얻은 Visual cue를 통해서 판단한 후에, 잠수함의 새로운 위치를 입력하기 위해 조이스틱을 조작한다. 조이스틱을 통해 입력되는 값들은 가속도, 조향각, 어뢰 버튼 등이 있다. 가속도와 조향각은 잠수함의 동역학 모델에 적용되어, 잠수함의 위치와 자세를 계산하는데 사용된

다. 어뢰 발사 버튼과 같은 이벤트 성격의 입력값은, 이벤트 처리기(Event handler)를 통하여 모션 커맨드로 사용된다. 새롭게 갱신된 잠수함의 새로운 위치와 자세 값들은, HLA/RTI를 통해서 가시화 모듈과 모션 제어 모듈에 전달된다.

3.2.2 모션제어 모듈

Fig. 2(b)에서 보는 바와 같이 모션제어 모듈은 시스템 제어 모듈에서 계산된 잠수함의 가속도, 기울기, 진동 이벤트 등을 입력값으로 받아서, 모션제어를 움직여 줌으로써 운동감을 생성하는 체계이다. 입력값을 이용하여 모션제어 모듈에서는 선형 가속도와 원심력을 계산하고, 이것을 워시아웃 필터(Washout filter)를 통해서 모션제어에 적용 가능하도록 변환한다. 또한, 진동이나 기울어짐과 같은 효과를 직접 추가하여, 새로 갱신된 값과 이전 값을 중첩(Superposition) 함으로써 운동감을 증가시킨다.

본 연구에서는 2자유도의 운동감(Roll, Pitch)을 표현해 주는 모션제어를 사용하였다. 이 모션제어는 직렬 COM1 포트와 4800 bps의 Baud rate으로 설정된 RS232C 직렬통신 프로토콜을 사용하였다. 이 모션제어를 구동하기 위해 네트워크 상에서 전송되는 패킷은 다음과 같다. 만약 전송되는 패킷의 형태가 40 4D FE 7F 0A 00 00 14이 되면, 제어는 현재의 위치에서 가장 빠른 속도로 정면으로 기울게 된다. 체크섬(Check sum)인 14는 앞의 7바이트들의 합을 나타내어서 신호가 제대로 들어왔는지 확인하는 숫자로서, 앞의 7바이트 합이 16진수로 214이기 때문에 14가 들어간다.

소규모 모션 플랫폼의 하나인 2자유도의 모션제어는, 물리적으로 Sway, Heave, Surge, Yaw의 움직임을 표현하지 못하기 때문에, 자유도의 감소로 인한 현실감의 저하를 보상해 주기 위하여 워시아웃 필터를 적용한다. 워시아웃 필터란 모션 플랫폼이 특정 운동감을 표현하지 못할 경우에, 모션 플랫폼의 움직임을 조정하는 각각의 액추에이터(Actuator) 운동 변위와 가속도를 제어하여, 운동감을 보상해 준다.

본 연구에서 사용된 2자유도의 모션제어는, 두 개의 액추에이터의 회전각을 조절하여, 모션 플랫폼의 기울어짐을 나타낸다. 이 기울어짐을 조합하여 원하는 운동감을 근사할 수 있는 워시아웃 필터를 만든다. 즉, 일반적으로 뉴턴의 운동법칙인 $f=ma$ 에 의해서 질량이 일정하다고 가정할 경우, 운동감을 가속도로 표현할 수 있다. 그러나 2자유도 모션제어는 회전운동만을 나타낼 수 있을 뿐, 선형운동은 나타내지 못

하기 때문에, 실제적으로 가속도를 표현할 수 없다. 이에 가속도와 유사한 운동감을 줄 수 있는 방법으로, 모션제어의 병진 운동이 아닌 기울어짐을 사용할 수 있다.

예를 들어 선형 가속도의 경우에, 자동차가 급정거를 하거나 급출발을 하면, 타고 있는 사람은 몸이 앞이나 뒤로 쏠리는 것을 느낄 수 있다. 그러므로, 앞뒤로의 가속을 나타내는 선형 가속도가 양의 방향으로 조금 작용한다면, 모션제어를 뒤 방향으로 조금씩 천천히 기울여 주어서 관성을 느끼게 할 수 있고, 선형 가속도가 크게 작용한다면, 모션제어를 뒤 방향으로 많이 기울여 주어서 큰 관성을 느끼게 할 수 있다. 즉, 선형 가속도의 크기는 모션제어를 앞 뒤로의 기울어짐으로 나타낸다.

회전에 의한 원심력을 생각해 보자. 놀이 공원에서 회전목마를 타는 경우에, 말을 타고 있는 사람은 몸이 바깥쪽으로 쏠리는 것을 느낄 수 있다. 그러므로, 회전에 의한 원심력을 표현하기 위해서, 모션제어를 옆으로 기울여 주면 된다. 즉, 원심력의 크기는 옆으로의 기울어짐으로 나타낸다. 원심력의 크기는 다음의 수식을 사용하였다.

$$\text{원심력} = m \frac{v^2}{r} \quad (1)$$

여기서, m : 탑승자의 질량

v : 잠수함의 진행방향 속도

r : 잠수함의 회전반경

워시아웃 필터와 관계없는 진동효과와 잠수함의 기울기는, 직접 모션제어에 전달하여 중첩시킴으로써 운동감을 증강시킬 수 있다(Fig. 2(b) 참조). 진동은 Heave 모션이 있어야 가능하지만, 운동판에 Heave 모션이 없을 경우에 Roll과 Pitch 모션을 임의적으로 주어서, 모션제어가 전후 좌우로 빠른 주기로 왔다 갔다 함으로써, 유사 진동을 줄 수 있다.

이렇게 구해진 선형 가속도, 원심력, 기울어짐, 진동은 모두 모션제어의 Roll과 Pitch의 각도로 대체 표현할 수 있고, 중첩 과정을 통해 최종 Roll과 Pitch의 변위각이 모션제어에 적용이 된다.

3.2.3 가시화 모듈

가시화 모듈은 시스템 제어 모듈을 통해 계산된 잠수함의 위치와 자세 값을 입력 받아서 화면에 가시화하는 체계이다. Fig. 2(c)에서 보는 바와 같이 전달된 잠수함의 위치와 자세값은 Multi-channel manager에

의해서 4개의 독립된 PC의 Image generator에 각각의 뷰포인트(Viewpoint) 정보와 함께 전달되어 다채널의 가시화가 가능해진다. 시뮬레이션 과정에서 연속된 장면에 대해 다채널의 가시화를 실현하기 위해서, 실시간으로 넘어 오는 시뮬레이션 결과값을 X3D 장면그래프(scene graph)에 반영하고, 이 장면그래프를 시뮬레이션 결과값에 맞게 변형시켜야 한다.

그러나 X3D포맷은 정적인 3D 형상 정보만을 가지고 있고, 장면그래프를 변경하기 위한 동적인 정보는 포함하고 있지 않다. X3D의 장면그래프를 동적으로 바꾸기 위해서 VRML의 EAI(External authoring interface) 기능에 대응되는 SAI(Scene access interface)를 사용한다. SAI는 Web3D 컨소시엄에서 X3D의 장면그래프를 동적으로 바꿀 수 있도록 제안한 일종의 API (Application programming interface)이다^[17].

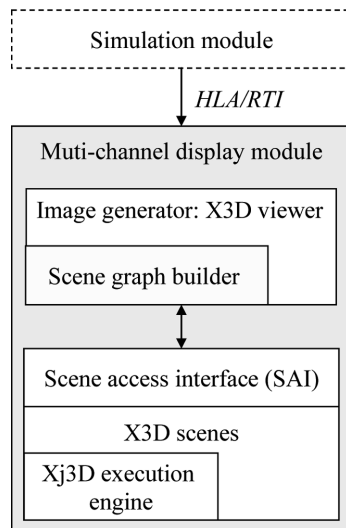


Fig. 3. Process to dynamically change X3D scene graph using SAI.

Fig. 3에서 보는 바와 같이 시스템 제어 모듈을 통해 계산된 잠수함의 위치와 자세 값을 HLA/RTI를 통해 전달받은 X3D 뷰어는, X3D 응용프로그램인 Scene graph builder를 통해 장면그래프로 변환한다. 즉, X3D 장면은 Xj3D 엔진을 통해서 화면에 가시화 되고, Scene graph builder는 SAI를 이용하여 X3D의 장면그래프를 실시간으로 바꾸어 준다.

3.3 다채널 디스플레이의 동기화

가상환경 공간에서 사용자에게 몰입감을 증대시키기 위하여, 하나의 장면을 여러 대의 PC를 이용하여 다수의 화면에 가시화하는 다채널 시스템을 사용한

다. 일부의 고가 상업용 시스템에서도 다채널 가시화를 지원하는 함수를 제공하지만, Multi-processor와 Multi-pipeline을 지원하지 않는 일반 PC를 사용할 경우에는, 네트워크 상에 있는 다수의 PC들을 클러스터링 한 후에 동기화 하는 방법이 필요하다.

본 논문에서는 각각의 PC에서 가시화를 위한 프로그램을 동시에 실행한 후, 각각의 프로그램을 동기화 하는 방법을 사용하였다. Fig. 4는 잠수함에 대한 장면에 대해, 하나의 채널에서 가시화 영역인 절두체 모양의 Viewing frustum이 두 개의 채널로 변경되었을 경우에 변화된 Viewing frustum을 보여준다. 두 개의 화면으로 하나의 장면을 나타내기 위해서는, 두 개의 Viewing frustum을 사전에 계산된 각도로 회전시켜 뷰포인트를 설정하는 것이 필요하다.

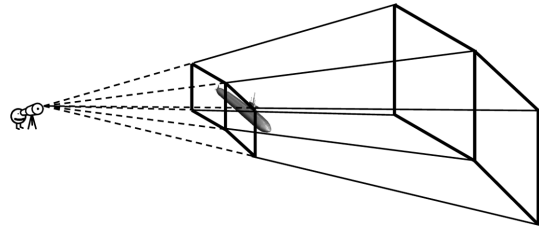


Fig. 4. Concept of synchronization in a double-channel display.

Viewing frustum을 적절한 각도로 회전시키기 위하여, Fig. 5에서 보는 바와 같이 하나의 화면이라고 가정하면, Aspect ratio는 화면의 가로와 세로의 비율이므로 다음과 같이 나타낼 수 있다.

$$\begin{aligned} \text{Aspect ratio} &= \frac{\text{resolution width}}{\text{resolution height}} \\ &= \frac{a}{b} = \frac{2l \times \tan \theta}{2l \times \tan \phi} = \frac{\tan \theta}{\tan \phi} \end{aligned} \quad (2)$$

PC의 가로와 세로에 대한 해상도는 이미 설정이 되어 있고, 스크린의 가로와 세로 길이 또한 알고 있기 때문에 Aspect ratio를 구할 수 있다. 즉, θ 와 ϕ 의 관계를 알 수 있다. 사용자와 화면 사이의 거리를 l 이라고 하면, 앞에서 구한 관계식에 의해서 θ 와 ϕ 를 다음과 같이 모두 구할 수 있다.

$$\begin{aligned} a &= 2l \times \tan \theta \\ \therefore \theta &= \tan^{-1} \frac{a}{2l} \\ \therefore \phi &= \tan^{-1} \frac{\tan \theta}{\text{Aspect ratio}} \end{aligned} \quad (3)$$

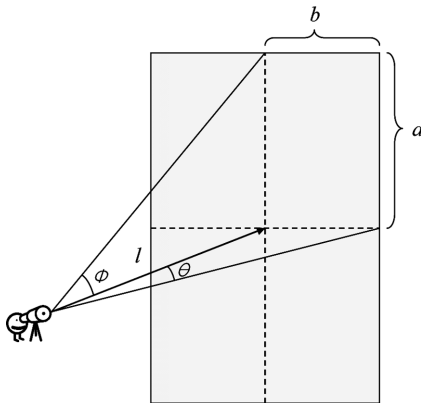


Fig. 5. Variables to derive conditions for synchronization of multi-channel display system.

따라서 FOV(Field of view, 시야각)는 식 (2)와 (3)에 의해서 계산된 θ 와 ϕ 가운데 작은 값의 2배이므로 다음과 같은 식으로 표현될 수 있다.

$$\text{Field of view} = 2 \times \min(\theta, \phi) \quad (4)$$

위에서 구한 FOV를 다채널 가시화 시스템에 적용시킨 후, 뷰포인트를 θ 만큼 회전시켜 주면 하나의 장면을 두 개의 화면으로 보여줄 수 있다.

클러스터 된 PC들 사이에 채널 간의 동기화를 실현하기 위해 네트워크 상에서 동기화 패킷을 각각의 PC에 전달하는 방식을 사용한다. 가시화 모듈 내에서 각각의 채널들은 물리적으로 근 거리에 위치하고 있기 때문에, RTI를 사용하는 대신 속도가 빠르고 직접적인 제어가 가능한 TCP(Transmission control protocol)나 UDP(User datagram protocol)를 사용할 수 있다. 그러나 동기화해 주어야 할 채널들이 많거나 초당 전달해야 할 패킷이 많을 경우에, TCP를 사용하면 속도가 느려질 수 있으므로 본 논문에서는 UDP를 사용한다.

3.4 Federate 상호간에 메시지 전달 체계

분산 시뮬레이션을 위한 Federate들의 구성이 완료되면 Federation이 실행된다. Federate 상호간에 메시지 전달 체계를 구축하기 위해 도입된 HLA의 FOM(Federation object model)은 전체 분산 시뮬레이션이 동작할 때에, Federation의 구성원인 Federate들 사이에 공유되어야 하는 정보에 대한 자료구조를 계층적으로 정의한 것이다. 예를 들어, 잠수함 Federate와 어뢰 Federate로 구성된 Federation이 있다고 가정하면, 잠수함 Federate와 어뢰 Federate 간에는 객체의 위치

정보를 서로 교환하여야 한다. 잠수함 객체, 어뢰 객체, 그리고 각 객체의 위치 속성 등이 바로 FOM에 표현된다. 3.2절에서 언급한 3개의 Federate 가운데 FOM에 포함되어야 할 객체는, 잠수함의 위치와 자세 정보를 가지고 있는 시스템 제어 모듈인 SM_Info이다. 나머지 두 개의 Federate인 모션채어 모듈의 인스턴스와 가시화 모듈의 인스턴스는, 시스템 제어 모듈이 송신(publish)하는 SM_Info의 인스턴스로부터 정보를 수신(subscribe)만 할 뿐, 모션채어 모듈과 가시화 모듈 사이에 정보 공유는 필요가 없다.

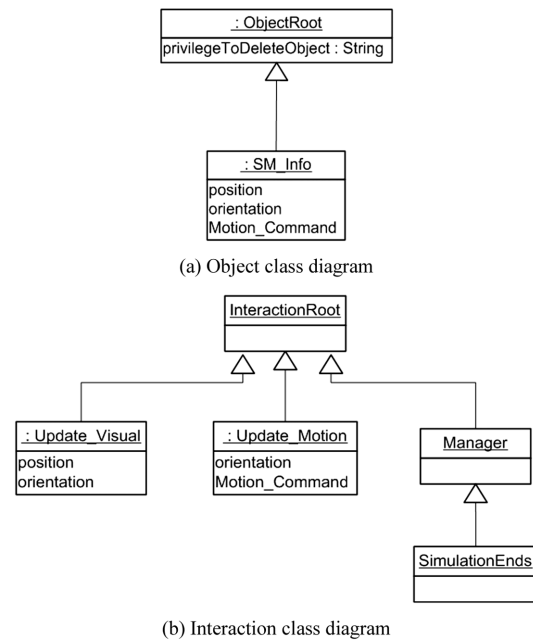


Fig. 6. UML diagram for object and interaction.

Fig. 6(a)는 FOM 중 객체 클래스와 그 속성들, 그리고 클래스 간의 상속관계를 UML(Unified modeling language)로 표현한 것이다. HLA의 모든 객체 클래스는 ObjectRoot 클래스로부터 상속받기 때문에 privilegeToDeleteObject의 속성을 갖는다. 이 속성을 상속받은 Federate들은 모두 객체 인스턴스를 소멸시킬 수 있다. 모션채어 객체와 가시화 모듈 객체는 따로 속성을 갖지 않는다. Fig. 6(b)는 FOM 중 Interaction 클래스의 구조를 보여준다. SimulationEnds 클래스는 시뮬레이션 종료를 명령할 때에 사용된다. SimulationEnds 클래스가 상속받은 Manager 클래스는, 시뮬레이션 관리를 목적으로 HLA에서 기본적으로 제공하는 클래스이다. 시스템 제어 모듈에서 SimulationEnds라는 Interaction을 송신하면, 이를 수

신한 모션체어와 가시화 모듈 Federate들은 동작을 종료한다.

Federation이 실행되면, Federation에 참여한 Federate들은 몇 가지 객체의 속성들을 설정하게 되는데, 본 논문의 경우, 시스템 제어 모듈의 Federate는 잠수함 객체의 위치, 자세 그리고 이벤트 모션과 같은 속성을 송신하며, 모션체어 모듈과 가시화 모듈의 Federate는 이렇게 송신된 속성들을 수신한다. 이때 설정된 송수신 관계는 RTI에 저장된다. Fig. 7은 Federation을 구성하는 3개의 Federate들 사이에 메시지 전달 과정을, UML의 Sequence diagram으로 나타낸 것이다. 시스템 제어 모듈에서 잠수함 객체의 속성(위치, 자세, 이벤트 모션)을 사용자 입력에 따라 업데이트 한 뒤, RTI의 명령어인 Update_Attribute_Values를 사용해서 송신하게 되면, RTI는 RTI에 저장된 송수신 관계에 검색하여, 업데이트된 속성값을 수신하기 원하는 Federate에게 RTI 명령어인 Reflect_Attribute_Values+를 사용해서 전달한다. 여기서 + 기호는 RTI가 직접 호출하는 Callback 함수이다.

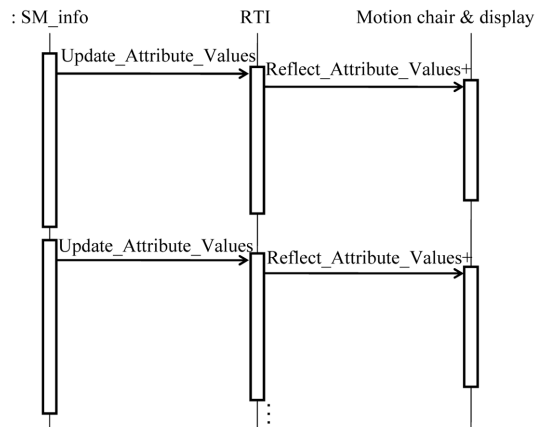


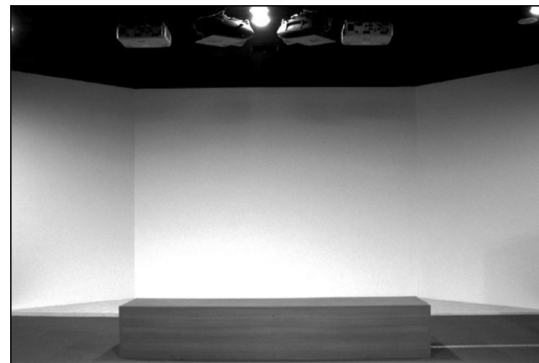
Fig. 7. Sequence diagram for transferring message between federates through RTI.

4. 시뮬레이터의 구현과 실험

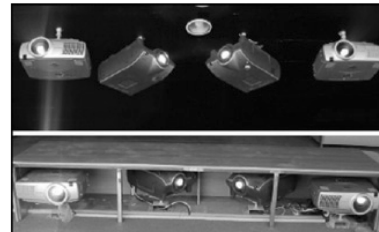
4.1 iCAVE 시스템

본 논문에서 제안된 모션체어와 X3D를 이용한 다채널 수중운동체 시뮬레이터는, Fig. 8에서 보는 바와 같이 한국과학기술원에 구축된 다채널 가상현실 시스템인 iCAVE를 기반으로 구현하였다^[20,23]. iCAVE 시스템은 1900×1400 mm 크기의 스크린 4개와, 범용 빔프로젝터 8대로 이루어져 있다. 두 개의 빔프로젝터가 한 개 스크린의 상하면에 각각 투사된다. 수중운

동체 시뮬레이션을 위하여 7대의 PC (시스템 제어 모듈 1대, 모션체어 모듈 1대 그리고 가시화 모듈 5대)가 설치되었다. 가시화 모듈을 위해 설치한 5대의 PC는 X3D 모델을 스크린에 출력하는 역할을 담당하는 디스플레이용 슬레이브(Slave) PC가 4대, 그리고 슬레이브 PC들이 출력하는 각각의 X3D 장면들을 하나의 큰 장면으로 생성될 수 있도록 동기화를 해 주는, 한 대의 마스터(Master) PC로 구성되었다. 슬레이브 PC의 그래픽카드는 두 대의 Projector와 연결되어 있다. 동기화를 위해서 마스터 PC는 100M Bps Ethernet으로 연결된 LAN을 통해서, 각각의 슬레이브 PC에 동기화 패킷을 전달한다. 각각의 슬레이브 PC는 1024×1536의 해상도를 가지므로, 전체 화면은 4096×1536의 해상도를 가진다.



(a) Four screens with 1900x2800 mm



(b) Eight front projection type projectors

Fig. 8. Virtual environment of iCAVE.

4대의 슬레이브 PC와 1대의 마스터 PC를 연결하는 네트워크 프로토콜은 UDP를 이용하였다. PC들 간에 네트워크를 연결하기 위해 일반적으로 사용되는 TCP는, 네트워크의 장애가 발생할 경우에 연결의 보장을 위해서 패킷을 재전송 하는 과정을 거치기 때문에 속도가 느려지는 단점이 있다. UDP는 다수의 PC들 간에 연결을 보장하지 않기 때문에, 패킷의 손실이 발생할 수 있고 전송되는 순서가 뒤바뀔 수 있다. 그러나 UDP는 패킷을 보낸 후 패킷에 대한 완전성을 신경

쓰지 않기 때문에, TCP/IP에 비해 속도 면에서 빠르다는 장점을 가진다. 수중운동체 시뮬레이션 시스템에서는 슬레이브 PC들과 마스터 PC가 근거리 네트워크 상에 존재하므로, UDP의 패킷 손실이나 도착 순서의 뒤바뀜 등은 거의 일어나지 않는다. 따라서 속도에서 우위를 가지는 UDP를 사용하였다.

4.2 구현환경

잠수함 시뮬레이터를 구현하기 위해서, X3D 포맷 기반의 다채널 디스플레이를 적용하였고, 이를 모션 플랫폼의 하나인 저가형 2자유도의 모션제어와 연동하여 운동감을 추가하였다. HLA프레임워크의 연동을 위하여 MAK社의 MAK RTI를 사용하였다. Table 1은 구현 환경에 대하여 보여준다.

Table 1. System environment

Software	Operating system	Windows XP SP2
	HLA/RTI	MAK RTI (C++)
	X3D SDK	Xj3D Toolkit (Java)
	Network protocol	Viewer side: UDP Federate: RTI
Hardware	VGA card	Nvidia GeForce 6600
	CPU	Intel P4, 3.0GHz
	RAM	1GB
	Input device	Microsoft SideWinder Precision 2 Joystick
	Virtual environment	8 channel iCAVE (Fig. 8)
	Motion chair	2 DOF Joychair ^[24]



Fig. 9. Result of integration of X3D multi-channel display, motion platform and HLA/RTI.

Fig. 9는 iCAVE 시스템을 기반으로 구현된 다채널의 수중운동체 시뮬레이터를 보여 준다. 이 시뮬레이

터는 잠수함의 가속과 감속이 가능하고, 여러 가지 뷰포인트를 설정할 수 있기 때문에, 다양한 각도에서의 잠수함 시뮬레이션을 실현할 수 있다. 또한 잠수함이 움직임에 따른 선형 가속도, 원심력, 잠수함의 기울어짐, 어뢰 폭발후의 진동을 모션제어를 통해서 표현할 수 있다.

4.3 실험

본 논문에서 구현된 다채널의 수중운동체 시뮬레이터에 적용할 시나리오에는 다음과 같다.

- 부산항 앞 바다에 잠수함 한 척과 적함 한 척이 멀리 떨어져 있다.
- 아군은 적군 잠수함의 위치를 파악한 후, 적이 어뢰의 사정 거리 안에 들어올 때까지 적군의 잠수함에 접근한다.
- 어뢰에 목표를 설정한 후 어뢰를 발사한다.
- 발사된 어뢰가 적함과 충돌 시 모션제어의 진동과 화면 떨림 발생한다.

위 시나리오에 대한 시뮬레이터의 실험 장면은 YouTube(<http://www.youtube.com/watch?v=PUBVsbVcuSE>)에 동영상상이 올려져 있다. 본 논문에서 제안한 방법과 고가의 상업용 툴인 Paradigm의 Vega, 그리고 중저가의 상업용 툴인 Right Hemisphere의 Deep Exploration CAD Edition에 대해, 4가지 항목(프레임율, 파일의 크기와 로딩 시간, 개방성, 이미지 품질)을 비교 평가를 하였다.

4.3.1 프레임율의 비교

Table 2는 시뮬레이션을 위한 가상의 공간을 구성하는 5개의 3D 형상(부산 내륙지형, 부산 해저지형, 잠수함, 어뢰, 부산 바다)에 대해, 카메라의 위치를 임의로 지정한 네 가지 뷰포인트(Viewpoint 1~4)에서의 프레임율을 비교한 것이다. 이 3D 형상들을 Xj3D를 이용해서 표현하는 경우와, Deep Exploration을 이용해서 표현하는 경우를 비교한 결과는, 첫 번째 뷰포인트(Viewpoint 1)을 제외하고는 비슷한 성능을 보였다. 다른 뷰포인트(Viewpoint 2~4)와 비교해서, Occlusion culling 기법에 의해 상대적으로 적은 개수의 폴리곤이

가시화되는 Viewpoint 1에서, Xj3D가 Deep Exploration 보다 프레임율이 높은 이유는, 공개된 프로젝트로 개발된 Xj3D가 성능이 우수하기 보다는, 기본적인 셰이딩이나 렌더링의 기능들에 초점을 두고 개발되었기 때문에 렌더링 속도가 빠른 것으로 판단된다. 폴리곤의 개수가 많았던 Viewpoint 2, 3, 4에서

Table 2. Comparison of experiment results

		Xj3D (X3D)	Vega (Open Flight)	Deep Exploration (Open Flight)
Frame per second	Viewpoint 1	30 fps	18 fps	11 fps
	Viewpoint 2	10 fps	12 fps	10 fps
	Viewpoint 3	8 fps	24 fps	8 fps
	Viewpoint 4	10 fps	24 fps	9 fps
Loading time		30 seconds	80 seconds	20 seconds
Openness	Cost	Free	More than \$20,000	\$1,495
	Platform	Platform independent	Win, Linux, SGI Irix (All versions for each operating system should be purchased)	Windows
Image quality		Objects at a distance are frequently flickering due to low precision (16bit) of z-buffer. However, it can be solved by separating boundary surfaces to some extent.		Objects at a distance are less frequently flickering due to relatively high precision (24bit or more) of z-buffer. Due to anti-aliasing, scenes are rendered with high quality at the cost of performance.

는 예상대로 Xj3D의 프레임율이 Deep Exploration에 비해 떨어졌다. 하지만, Vega의 경우에는 프레임율이 Xj3D에 비해 상대적으로 높았다. 그 이유는 Vega는 내부적으로 가변적인 LOD(Level of detail)를 적용하여, 멀리 있는 형상에 대해서는 화면에 표시하지 않음으로써, 한 프레임당 보여 주어야 하는 폴리곤(Polygon)의 개수를 줄였기 때문이다.

4.3.2 파일의 크기와 로딩 시간의 비교

Table 2에서 보는 바와 같이 Xj3D는 30초, Vega는 80초, Deep Exploration은 20초의 로딩시간을 갖는 것을 확인할 수 있다. 로딩 시간은 파일의 크기와 연관이 있다. Table 3에서 보는 바와 같이 3개의 파일 포맷에 대해 동일한 3D 형상의 크기를 비교하면, X3D가 VRML이나 Deep Exploration의 Open Flight보다 파일 크기가 더 작다는 것을 알 수 있다. X3D는 VRML과는 다른 Encoding 방식을 사용하기 때문에, X3D가 VRML보다 더 최적화된 파일 구조를 가지는 것을 알 수 있다. 그러나 Open Flight의 경우, X3D가 Open Flight보다 더 최적화된 파일 구조를

가진다고 말하기는 어렵다. 비록 바이너리(Binary) 형태의 X3D 파일 크기는 Open Flight보다 작지만, Open Flight는 내부적으로 X3D보다 더 많은 비형상 정보를 가지고 있기 때문이다. Deep Exploration은 내부적으로 최적화된 로딩 알고리즘을 사용하기 때문에, 동일한 3D 형상의 Open Flight 파일을 읽어 들이는데 걸리는 시간이 Vega보다 더 작다.

4.3.3 개방성 비교

X3D는 2003년에 ISO 국제표준으로 채택되었고, 개방된 프로젝트 형태로 지속적인 개발이 이루어지고 있다. Xj3D 뷰어와 같은 X3D를 가시화하는 응용프로그램들은, 별도의 비용이 없이도 사용이 가능하다 (Table 2 참고).

X3D는 플랫폼에 제약 없이 사용할 수 있는 반면에, Vega는 Windows, Linux, SGI Irix 등과 같은 이기종 플랫폼에 종속적이다. 따라서 서로 호환이 어렵고, 각각의 플랫폼에 대해 별도의 비용을 지불해야 하는 단점이 있다. 또한 Deep Exploration은 Windows 환경에서만 사용이 가능하다.

4.3.4 가시화 이미지의 품질 비교

Table 2의 이미지 품질 항목에 대한 결과에서 보는 바와 같이, 특정 장면에서 Xj3D와 Vega의 가시화 이미지가 깜박임(Flickering)이 발생하였다. 깜박임이 발생하는 장면은 주로 바닷물과 육지가 겹쳐져 있는 곳에서 일어난다. 또한 가까이서 볼 때 보다는 원거리에서 바라볼 때 이러한 현상이 나타난다. 그 이유는 물과 육지의 높낮이의 차이가 조금밖에 나지 않기 때문에, 멀리서 바라보게 될 경우 전체적인 가시화 스케일

Table 3. Comparison of file size (unit: Kbyte)

3D models	Xj3D(X3D)		VRML	Open Flight
	ASCII	Binary	ASCII	Binary
Pusan terrain	14,887	1,347	27,395	12,598
Pusan underwater terrain	939	144	2,602	1,222
Submarine	386	64	1,116	407
Torpedo	25	6	73	N/A
Pusan sea surface	22	5	35	83

이 커져서, 상대적으로 육지와 바다물의 높이 차이가 작아지게 된다. 이 높이의 차이가 그래픽 하드웨어의 z-buffer에 주어진 Depth의 차이에 대한 정밀도 보다 더 작게 되면, 그래픽 하드웨어는 두 개의 면 즉, 육지와 바다가 모두 같은 높이에 있는 걸로 착각을 하게 되어서, Hidden Surface Removal 또는 Occlusion Culling을 못하게 된다. 이로 인해 육지와 바다가 번갈아 가면서 보이기 때문에 깜박임 현상이 일어나는 것이다. 반면 Deep Exploration의 경우 Xj3D나 Vega에 비해 깜박임이 상대적으로 적게 발생하였다. 이것은 Deep Exploration이 Xj3D나 Vega보다 더 높은 z-buffer의 정밀도를 사용하기 때문이다. 깜박임 문제를 해결하기 위해서는 정밀도를 높이는 것이 해결책이지만, 해당 시스템의 개발자가 Vega의 정밀도를 높이거나 Xj3D의 소스코드를 수정하지 않는 이상은 해결하기 어려운 부분이다. 깜박임을 줄일 수 있는 방법은 육지와 바다의 높낮이 차이가 많이 나도록, 육지를 높이는 지 해수면을 낮추는 것이 있다. 실제로 해수면의 높이를 낮추어서 실험을 수행한 결과 깜박거림이 사라지는 것을 알 수 있었다.

Vega에는 Vega Marine이라는 모듈을 포함하고 있는데, 이 모듈을 사용하면 바다의 파도가 물결치는 것을 나타낼 수 있고, 바다 속의 모습도 좀 더 실제적으로 나타낼 수 있다. 본 논문에서는 바다물의 투명도를 조절하고 내륙지형, 해저지형 그리고 바다에 안개효과를 적용하여, 실제의 환경과 비슷한 효과를 낼 수 있었다.

5. 결 론

기존 수중운동체 시뮬레이터는 고가의 상업용 가상화 툴과 전용 하드웨어를 사용함으로써, 시스템을 관리하기 위한 특별한 장소의 문제, 유지 보수의 문제, 정해진 시간과 장소에서의 사용문제 등의 이유로 활용도가 낮다. 또한 단일 채널의 가상화 방법을 사용하고 있기 때문에 사용자의 몰입감이 떨어진다. 그리고 서로 다른 환경에서 만들어진 시뮬레이션들을 통합하기 어려운 문제도 발생하였다.

이러한 문제점들을 해결하기 위해서 본 논문에서는, 플랫폼 독립적이고 개방된 국제 표준인 X3D를 이용하여, 일반 사양의 PC들을 클러스터링하여 잠수함 시뮬레이션에 적용하였고, 몰입감 증대를 위해서 다채널 가상화와 모션 플랫폼을 적용하였다. 또한, 이들 모션채어, 조이스틱, 다채널 가상화 등 다른 환경에서 개발된 Federate들을 효과적으로 통합하기 위해

서 HLA/RTI를 적용하였다.

구현된 수중운동체 시뮬레이터를 이용하여 부산항 환경과 유사한 가상의 환경을 구축하여 실험을 하였다. 고가의 상업용 가상화 툴과 장비를 활용한 시스템과, 본 연구에서의 구현한 수중운동체 시뮬레이터를 비교해 볼 때, 성능 상에 있어서 큰 차이를 보이지 않음을 알 수 있었다. 또한 다채널 가상화와 모션 플랫폼을 적용함으로써, 실제 환경과 같은 몰입감을 증대시킬 수 있다.

본 논문에서는 한 대의 잠수함을 적용한 간단한 분산 시뮬레이션 환경을 구축하였다. 향후에는 여러 대의 잠수함과 비행기 및 전함 등을 배치하고, 실제로 교전을 하는 대규모의 시스템을 구축함으로써, HLA의 수행 능력이나 현실적인 문제점들을 파악할 수 있을 것이다.

마지막으로, 본 논문에서는 가상화 부분에 있어서는 공개된 국제표준을 사용하여 저가의 시스템을 구축할 수 있었지만, 분산 시뮬레이션 환경을 구성하는 프레임워크로서 상업용 MAK RTI를 사용하였다. 최근 XMSF(Extensible modeling and simulation framework)에서는 Web 기술을 적용하여 Web-Enabled RTI를 제안하고 있으며, 활발한 연구가 진행 중에 있다^[9]. 따라서 향후에는 MAK RTI 대신 Web-Enabled RTI를 사용함으로써, 분산 시뮬레이션 환경을 구성하는 프레임워크도 저가의 시스템으로 구성이 가능할 것이다.

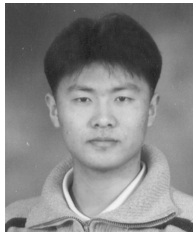
감사의 글

본 연구는 수중운동체특화연구센터의 ‘SM-11 수중운동체의 체계/부체계 기능 및 성능 시뮬레이션을 위한 네트워크 기반의 가상 복합 운동체 모델구조 연구’의 연구결과로 수행 되었습니다.

참고문헌

1. Brutzman, D., Zyda, M., Pullen, M. and Morse, K., “XMSF Overview, Progress Examples and Exercise Planning”, XMSF JFCOM Workshop, Virginia, USA, 2003. (<https://www.movesinstitute.org/xmsf/xmsf.html#JfcomWorkshopMay2003>)
2. Christianson, B., “Comparison of Vega and Java3D in A Virtual Environment Enclosure”, Master’s thesis, Naval Postgraduate School, March 2000.
3. Salisbury, C. and Farr, S., “Web-Based Simulation visualization Using Java3D”, Proceedings of the 31st Conference on Winter Simulation, pp. 1425-1429,

- 1999.
4. Li, Y., Brodrie, K. and Phillips, N., "Web-based VR Training Simulator for Percutaneous Rhizotomy", in *Medicine Meets Virtual Reality 2000* edited by JD Westwood, HM Hoffman, GT Mogel, RA Robb and D Stredney, IOS Press, pp. 175-181, 2000.
5. Robert, J. and Knight, R., "Multiple Window Visualization on the Web Using VRML and the EAI", *Proceedings of the 7th UK VR-SIG Conference*, pp. 149-157, 2000.
6. 문홍일, "SEDRIS를 이용한 디지털 생산 시뮬레이션 환경의 융합", *한국시뮬레이션학회논문지*, 제14권, 제2호, pp. 15-24, 2005.
7. Blais, C., Brutzman, D., Horner, D. and Nicklaus, S., "Web-Based 3D Technology for Scenario Authoring and Visualization: The SAVAGE Project", *Proceedings of the 2001 Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*, Orlando, Florida, 2001.
8. Blais, C., Brutzman, D., Weekley, J. and Harney, J., "Emerging Web-based 3D Graphics for Education and Experimentation", *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2002*, Florida, December 2002.
9. Sims, E., "Reusable, Lifelike Virtual Humans For Mentoring and Role-Playing", *Computers & Education*, Vol. 49, No. 1, pp. 75-92, 2007.
10. 이동훈, "X3D 가상환경에서의 확장 가능한 상호작용", 제30회 한국정보과학회 춘계 학술대회 논문집, pp. 349-351, 2003.
11. 이성태, "X3D를 이용한 Humanoid 모델링과 애니메이션 기법에 관한 연구", *한국해양정보통신학회 추계종합학술대회지*, pp. 812-816, 2002.
12. Singhal, S. and Zyda, M., "Networked Virtual Environments: Design and Implementation", Addison-Wesley, New York, 1999.
13. Judith, D. and Frederick, K., "Creating Computer Simulation Systems - An Introduction to the High Level Architecture", Prentice Hall, 1999.
14. Ping, I., "High Level Architecture Performance Measurement", Master's thesis, Naval Postgraduate School, 2000.
15. Jack Ogren, "EAGLE and the High Level Architecture", DMSO.
16. Defense Modeling and Simulation Office: High Level Architecture, <https://www.dmsomil/public/transition/hla/>, 2007.
17. The Xj3D Project, <http://www.xj3d.org>, 2007.
18. The Web3D Consortium: X3D Specification, <http://www.web3d.org/specifications/x3d-3.0.dtd>, 2007.
19. Extensible Modeling and Simulation Framework (XMSF), <https://www.movesinstitute.org/xmsf/xmsf.html>, 2007.
20. Kim, Y., Yang, J. and Han, S., "A Multichannel Visualization Module for Virtual Manufacturing", *Computers in Industry*, Vol. 57, No. 7, pp. 653-662, 2006.
21. 허필원, "X3D 다채널 가시화와 모션 플랫폼을 이용한 수중운동체 시뮬레이션의 HLA 기반 통합", 석사학위논문, KAIST, 2006.
22. 허필원, 한순홍, "수중운동체 시뮬레이션의 인터넷 기반 X3D 가시화", *한국시뮬레이션학회 2006년 춘계학술대회논문집*, pp. 48-58, 2006.
23. 김용식, 양정삼, 한순홍, "가상 공장 시뮬레이션을 위한 PC 클러스터 기반의 멀티채널 가시화 모듈의 설계와 구현", *대한기계학회논문집A*, 제30권, 제3호, pp. 231-240, 2006.
24. Contech Inc.: Joychair simulator, <http://www.contechvr.com/korean/product/jc.html>



허 필 원

2004년 서울 한양대학교 기계공학부 학사
2006년 한국과학기술원 기계공학과 석사
2006년~현재 University of Illinois at Urbana-Champaign, Department of Mechanical Science and Engineering, Human Dynamics and Control Lab, PhD Candidate

관심분야: Human postural control, Effect of visual information on the stability of postural control, Application of virtual reality (VR) to Biomechanics



한 순 홍

한국과학기술원 기계공학과 교수이며, 웹저널인 International Journal of CAD/CAM(www.ijcc.org)의 편집장으로 활동하고 있다. 2003년까지 STEP센터(www.kstep.or.kr)의 회장과 전자거래학회(www.calsec.or.kr)의 회장을 맡았으며, 관심분야는 STEP, 가상현실 응용, 지능형 CAD이다. 연락처는 shhan@kaist.ac.kr, 홈페이지 <http://icad.kaist.ac.kr>, 미국 미시건 대학에서 1990년 박사학위



양 정 삼

1991년 충남대학교 기계공학과 학사
1997년 한국과학기술원 기계공학과 석사
2004년 한국과학기술원 기계공학과 박사
1997년~2000년 고등기술연구원 주임연구원
2002년 Clausthal University of Technology (Germany) Visiting scholar

2001년~2005년 (주)부품디비 연구개발팀장
2005년~2006년 University of Wisconsin-Madison Postdoctoral associate
2006년~현재 아주대학교 산업정보시스템공학부 조교수
관심분야: Product data quality (PDQ), Product data exchange (PDE), Product data management (PDM), Geometric modeling, Virtual manufacturing