석 사 학 위 논 문
Master's Thesis

X3D 다채널 가시화와 모션 플랫폼을 이용한 수중운동체 시뮬레이션의 HLA 기반 통합

HLA-based Integration of Underwater Vehicle Simulations using X3D Multi-Channel Visualization and a Motion Platform

허 필 원 (許 弼 援 Pilwon Hur)

기계공학과 기계공학전공
Department of Mechanical Engineering
Division of Mechanical Engineering

한 국 과 학 기 술 원
Korea Advanced Institute of Science and Technology

2006

X3D 다채널 가시화와 모션 플랫폼을 이용한 수중운동체 시뮬레이션의 HLA 기반 통합

HLA-based Integration of Underwater Vehicle Simulations using X3D Multi-Channel Visualization and a Motion Platform

HLA-based Integration of Underwater Vehicle Simulations using X3D Multi-Channel Visualization and a Motion Platform

Advisor : Professor Soonhung Han

By

Pilwon Hur

Department of Mechanical Engineering
Division of Mechanical Engineering
Korea Advanced Institute of Science and Technology

A thesis submitted to the faculty of the Korea Advanced Institute of Science and Technology in partial fulfillment of the requirements for the degree of Master of Engineering in the Department of Mechanical Engineering, Division of Mechanical Engineering.

Daejeon, Korea
2006. 6. 30.
Approved by

_____

Professor Soonhung Han
Major Advisor

X3D 다채널 가시화와 모션 플랫폼을 이용한 수중운동체 시뮬레이션의 HLA 기반 통합

허 필 원

위 논문은 한국과학기술원 석사학위논문으로

학위논문심사위원회에서 심사 통과하였음.

2006년 6월 30일

심사위원장 한 순 흥 (인)

심사위원 권 동 수 (인)

심사위원 김 정 (인)

MME 20043998 허필원. Pilwon Hur. X3D 다채널 가시화와 모션 플랫폼을 이용한 수중운동체 시뮬레이션의 HLA 기반 통합. HLA-based Integration of Underwater Vehicle Simulations using X3D Multi-Channel Visualization and a Motion Platform. Department of Mechanical Engineering / Division of Mechanical Engineering. 2006. 83p. Advisor Prof. Soonhung Han

ABSTRACT

Submarine is very 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&Ss of submarine generally use very expensive commercial software and dedicated hardware, those systems should be kept in secure places, which cause the warfighters to take a lot of time to go to the places, and then to train themselves during very limited time. Another words, those systems are not effective. Also, many M&Ss have only one-channel display system which reduces immersiveness. Another problem is that many heterogenous 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 platform are used. Finally, HLA/RTI is used to integrate heterogenous simulators. All of these are verified through experiment.

목차 (Contents)

그림 (Figure) 목차

표 (Table) 목차

술어 및 용어 해설 (Nomenclature)

API	: Application Programming Interface
CAVE	: Computer Aided Virtual Environment
CG	: Computer Graphics
DDI	: Device Drive Interface
DIS	: Distributed Interactive Simulation
DMSO	: Defense Modeling and Simulation Framework
EAI	: External Authoring Interface
FOM	: Federation Object Model
FOV	: Field Of View
GDI	: Graphic Device Interface
GLU	: Graphic Library Utility
GUI	: Graphic User Interface
HLA	: High Level Architecture
M&S	: Modeling and Simulation
N/A	: Not Available
N/W	: Network
OSG	: Open Scene Graph
OMT	: Object Model Template
RTI	: Run Time Infrastructure
SAI	: Scene Access Interface
SDK	: Software Development Kit
SEDRIS	: Synthetic Environmental Data Representation and Interchange Specification
sgi	: Silicon Graphics Inc.
SOM	: Simulation Object Model
SVG	: Scalable Vector Graphics
S/W	: Software
SIMNET	: SIMulation NETwork
TCP	: Transmission Control Protocol

UDP	: User Datagram Protocol
UML	: Unified Modeling Language
VGA	: Video Graphics Array
VRML	: Virtual Reality Modeling Language
X3D	: Extensible 3D
XML	: Extensible Markup Language
XMSF	: eXtensible Modeling and Simulation Framework

1. 서론

1.1. 연구의 배경 및 동기

잠수함과 무인잠수정 및 잠수함에 탑재된 무기체계는 제2차 세계대전 이후 수상 및 수중 표적을 겨냥한 기존의 무기체계 개념에서 벗어나, 대지 육상 공격 능력을 갖는 수중발사 전략유도무기가 잠수함에 채택됨에 따라서 독립된 무기체계로서의 역할이 강조되어 왔으며, 80년대 초 포클랜드 해전 및 91년 걸프전을 통하여 은밀성과 그에 따른 위협성으로 인하여 대육상전 및 대함전 등을 통하여 어떠한 무기체계보다도 효과적이고 위협적인 방어 및 공격을 수행할 수 있는 무기체계임이 증명되었다. 특히 수중발사 전략 유도무기의 보유는 적의 선제공격에 대한 취약성을 최소화하고 전쟁 억제력 및 보복력을 극대화할 수 있는 무기체계로서 선진국을 중심으로 활발한 개발 및 연구가 진행되고 있음은 너무나도 잘 알려진 사실이다.

이러한 무기체계인 잠수함을 전쟁에서 잘 활용하는 것이 전쟁에서의 승패에 큰 영향을 미친다. 그러므로 장병들에게 잠수함을 잘 운용하는 방법을 교육시키는 훈련 또한 중요하다. 장병들에게 잠수함 사용법을 훈련시키는 방법에는 여러 가지가 있다. 그 중에 한가지는 실전과 같이 바다 속에서 잠수함을 직접 타고 훈련을 하는 방법이다. 이 방법은 실제 상황에서 똑같은 소리와 진동을 느끼면서 직접 장비를 조작할 수 있기 때문에 가장 큰 훈련의 효과가 있다. 하지만, 잠수함이라는 장비가 아주 고가이고, 바다라는 지역적 한계, 그리고 한번 훈련을 나갈 경우 최소 몇 개월이라는 시간적 한계 때문에 그렇게 실용성 있는 훈련 방법은 아니다. 다른 방법으로는 잠수함의 환경을 비슷하게 꾸미고 실제로 잠수함의 움직임을 모사하는 M&S(Modeling and Simulation) 방법이다.

M&S에 관해서 설명하기 전에 간단한 예를 들어보자. 해군에서 새로 개발한 잠수함이 수중에서 얼마나 빠른 속도로 움직일 수 있는지를 알고 싶을 경우, 실재로 잠수함을 물 속에 집어 넣고 최대의 마력으로 추진시켜 이때의 잠수함의 속도를 측정하면 된다. 이런 경우에는 M&S가 크게 필요하지 않을 것 같다. 이번에는, 잠수함을 향하여 어뢰가 발사되었을 경우, 어느 정도의 확률로 어뢰를 회피하여 생존할 수 있는가를 알고 싶다고 하자. 이런 경우에는 실재 잠수함을 타고 실험을 할 수 없다. 이때 사용할 수 있는 방법으로 모의 실험 또는 가상 실험 등이 있다. 이러한 실험을 시뮬레이션이라고 한다.

시뮬레이션을 수행하기 위해서는 앞에서 예로 든 잠수함과 비슷하게 행동하는 가상의 시스템이 필요하다. 이 시스템을 모델이라고 하고, 모델을 생성하는 과정을 모델링이라고 한다.

이러한 M&S 기법을 사용하면 저비용으로 지역적 및 시간적 한계를 극복하여 잠수함 군사훈련을 수행할 수 있기 때문에 사실감에서 조금의 불이익이 있어도 많이 사용되고 있는 방법이다[34].

1.2. 문제점 및 연구 내용

하지만, 정작 이렇게 만들어 놓은 시뮬레이션 시스템은 실질적으로 장병들의 잠수함 전술 운용 훈련에 별로 도움이 되지 못하는 경우가 있다고 한다[44]. 그 이유로는 잠수함 M&S의 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. 이러한 이유로 장병들은 잠수함 M&S를 접하는 것에 한계가 있으며, 거리감을 느끼기 쉽다[44].

단일채널의 가시화 방법은 일반적으로 많이 사용하는 방법이다. 하지만, 여러 개의 채널을 사용함으로써 몰입감을 높일 수 있다. 여기에 모션 플랫폼을 추가할 경우, 운동감을 추가하여 더욱 더 몰입감을 높일 수 있다. [Fig. 1-1]은 국내의 한 업체에서 고가의 상업용 툴과 단일채널을 사용하여 개발한 잠수함 M&S 결과를 나타낸 것이다.

일반적인 N/W(Network) 프로토콜 또는 프레임워크를 사용할 경우 서로 다른 상황과 환경에서 만들어지고 운용되는 시뮬레이터들을 통합하기 어려운 문제도 발생한다. 그렇기 때문에 육군과 공군을 포함하거나 다른 나라의 군사들을 포함하는 등 훈련의 규모가 확대되는 것에 장애가 된다.

그러므로 본 연구에서는 첫째 고가이고 전용시스템에 의존적인 기존의 잠수함 M&S 가시화 대신 일반적으로 쉽게 사용할 수 있는 저가이고 플랫폼 독립적인 가시화 시스템을 구성하고, 둘째 단일채널의 가시화 방법 대신 다채널의 가시화 및 모션 플랫폼을 적용하여 몰입감을 높이고, 마지막으로 서로 다른 환경의 시뮬레이터들도 통합이 가능한 프레임워크를 적용함으로써 시뮬레이터들의 호환성과 재사용성을 증대시킬 수 있는 시스템을 통합하고자 한다.

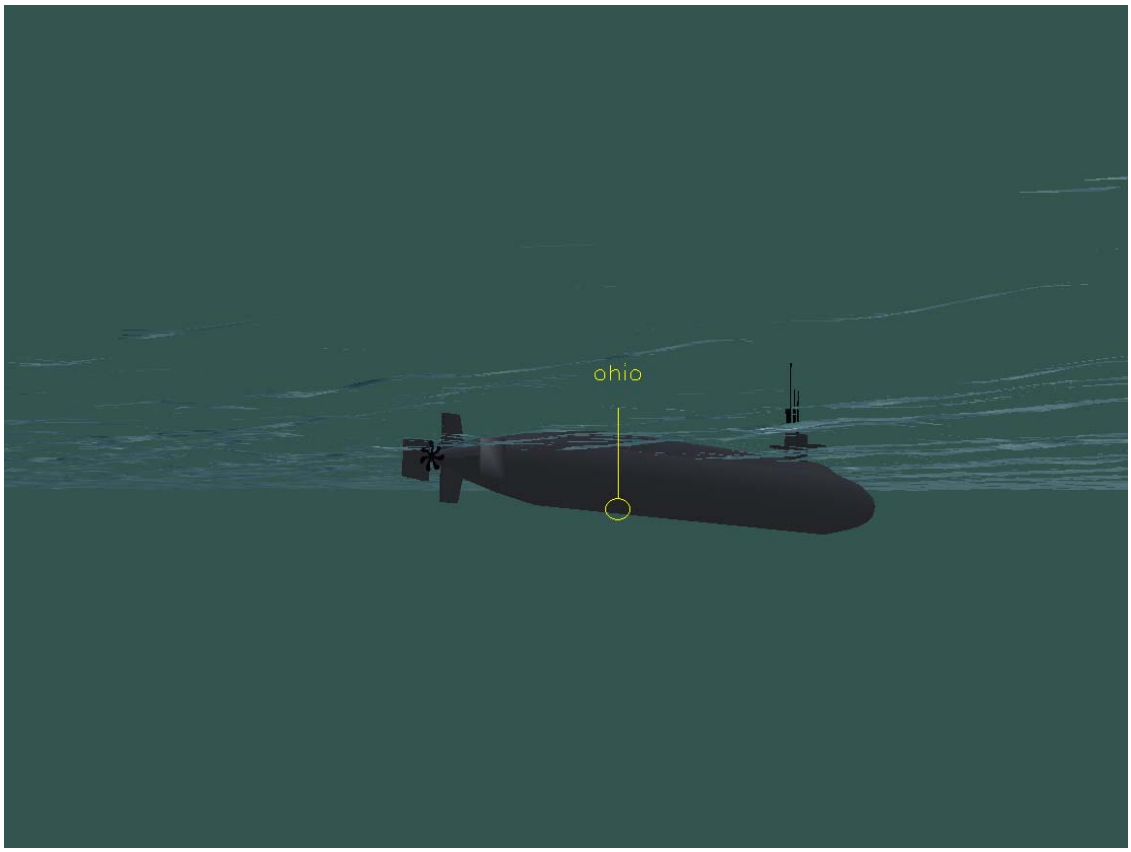

Fig. 1-1 고가의 시스템을 사용한 잠수함 M&S의 예

본 논문의 구성은 다음과 같다.

2장에서는 컴퓨터 그래픽스에 대한 간단한 분류에 대해서 소개한다. 그리고 기존 연구의 가시화 방법 및 포맷을 비교 및 분석한 후, 플랫폼 독립적이며 저가형인 가시화 시스템 구축을 위해서 본 연구에서 사용할 적절한 가시화 방법 및 포맷을 제시한다. 또한 기존에 사용되던 분산 시뮬레이션 프레임워크에 관해서도 분석한다.

3장에서는 본 연구에서 사용할 저가형 가시화 시스템에 적용할 공개된 국제표준 파일 포맷인 X3D(eXtensible 3D)에 관해서 소개한다.

4장에서는 서로 다른 환경의 시뮬레이터들을 통합할 수 있는 프레임워크인 HLA(High Level Architecture)에 관해서 설명하고 HLA/RTI(HLA/Run Time Infrastructure)를 통해서 각 Federate 들 간에 데이터가 이동하는 방법에 대해 소개한다.

5장에서는 다채널 가시화 및 모션 플랫폼에 대해서 소개한다.

6장에서는 앞의 2, 3, 4, 5장에서 언급한 방법들을 통합한 시스템에 대해서 설명하고 구현 및 실험결과를 소개한다.

마지막으로 7장에서는 본 연구의 내용을 요약하고 연구의 기여도와 X3D 국제 표준의 개선 요구사항을 설명하며, 향후 연구에 대해서 소개한다.

2. 관련연구

2.1. 가시화의 방법

2.1.1. 3D 그래픽스의 배경 및 분류

최근 들어 웹 상에서의 3차원 가시화에 관한 말을 많이 한다. 이와 관련해서 자주 듣는 용어 중 몇 가지로, Flash, SVG, VRML, Java3D, X3D 와 같은 것들이 있다. 단지 웹 상의 가시화가 아니라도 다음과 같은 용어들 즉, OpenGL, Direct3D, Scene Graph 등을 많이 듣게 된다. 여기에 나온 용어들은 모두 2D, 3D 가시화에 사용되는 용어들이지만, 몇 가지 구분이 필요하다. 이는 [Fig. 2-1] 에서 보는 바와 같이 구분할 수 있다. 2D는 관심분야가 아니므로 간략히 파일 포맷으로만 나타내었고, 여기서는 아래와 같은 분류에 의해서3D 부분을 중점적으로 살펴보자.

- Low Level Graphic Library (or API)
- High Level Graphic Library (or API)
- Graphic File Format

각각에 대해서 설명을 해 보면 다음과 같다.

2.1.1.1 Low Level Graphic API

저 수준의 그래픽 라이브러리로, DDI(Device Drive Interface)를 거쳐 그래픽 하드웨어에 접근하게 된다. 주로 3D 이미지를 렌더링 하는 것에 사용되며 OpenGL이나 Direct3D가 여기에 해당한다. [Fig. 2-2]와 [Fig. 2-3]는 각각 Direct3D와 OpenGL의 계층구조를 나타낸 것이다. 둘 다 비슷한 구조를 가지는 것을 알 수 있다.

OpenGL은 sgi社의 IRIS GL 라이브러리 후속으로 나온 그래픽 라이브러리이다. IRIS GL은 SGI 워크스테이션에서만 사용 할 수 있었던 강력한 그래픽 API로, 개방된 표준이 필요함을 느낀 SGI는 OpenGL을 발표하였다. 하드웨어에 아주 가깝게 설계되어서, 렌더링의 질이 높다.

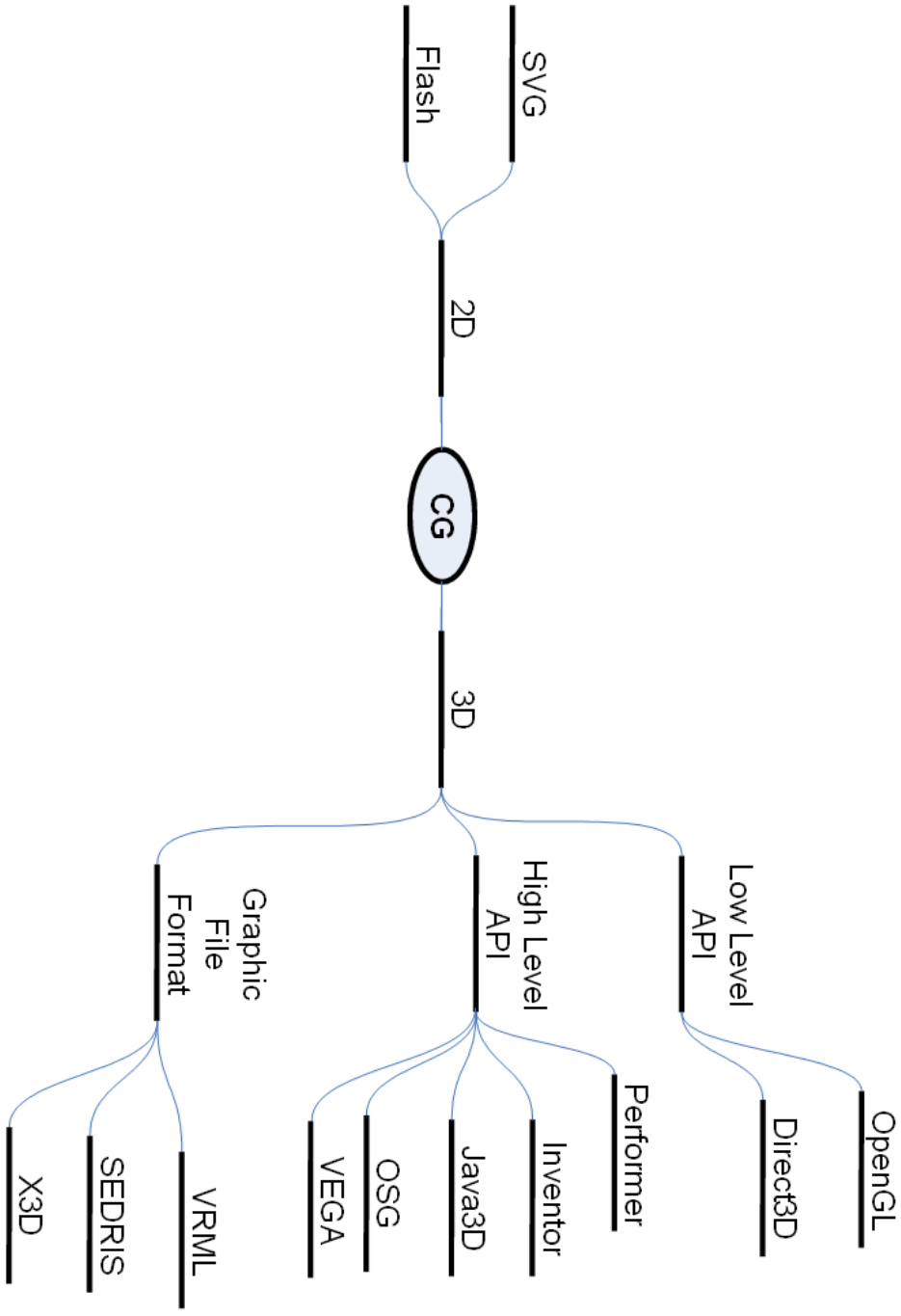

Fig. 2-1 컴퓨터 그래픽스의 구분

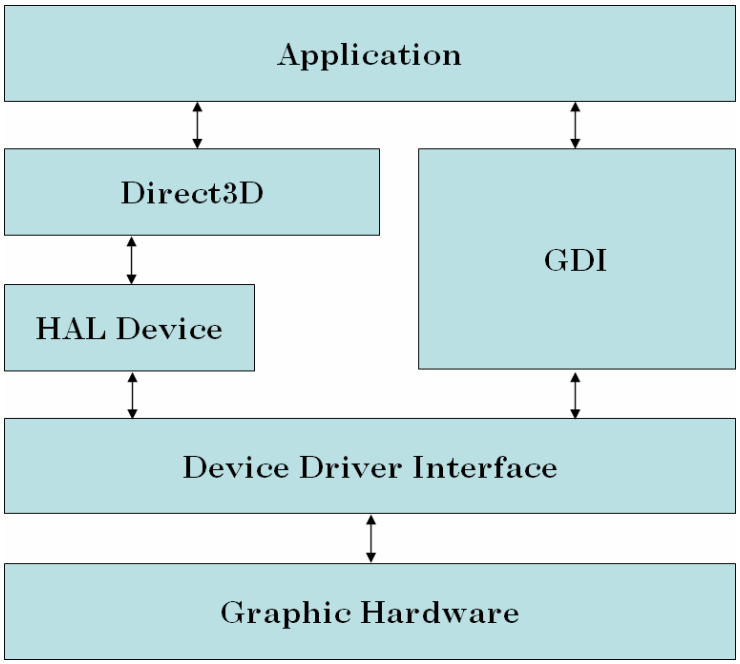

Fig. 2-2 Direct3D의 계층구조[8]

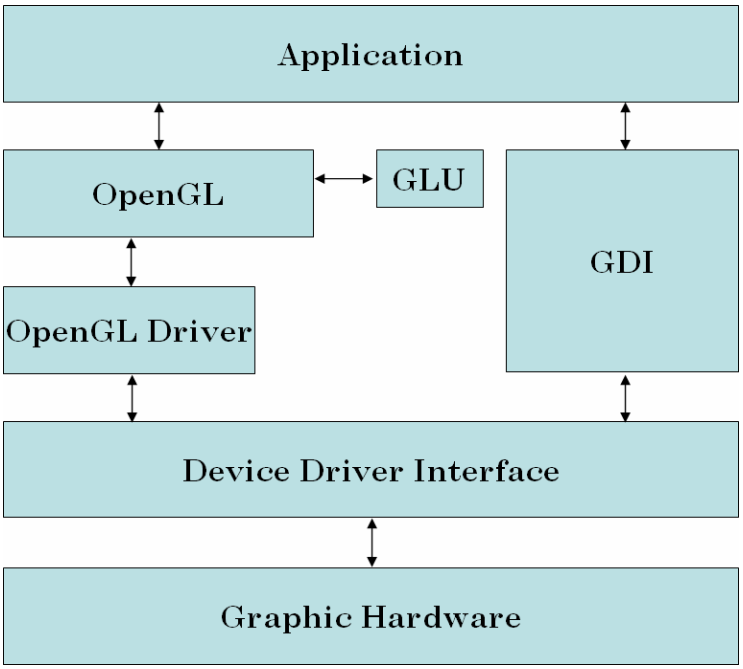

Fig. 2-3 OpenGL의 계층구조[9]

Direct3D는 마이크로소프트 사에서 1996년에 발표한 그래픽 라이브러리이다. 처음에는 윈도우에서 멋지고 고성능의 게임을 실행하기 위해 만들어 졌다. DirectX2.0 (1996), DirectX3.0 (1996), DirectX5.0 (1997), DirectX6.0 (1998), DirectX7.0 (1999), DirectX8.0 (2000), 그리고 DirectX9.0 (2003) 와 같이 발전을 해오면서 성능적으로 많이 향상 되었다. Direct3D는 마이크로소프트사의 윈도우 운영체계에서만 돌아가도록 설계되었기 때문에, 윈도우 환경에 최적화 되어 있다. 그렇기 때문에, 윈도우 환경에서 Direct3D는 OpenGL 보다 더 고속으로 렌더링 할 수 있다. 또한 Direct3D는 OpenGL 보다 행렬이나 벡터 기능을 잘 지원하고 있어 사용하기 편리한 점이 많다. 하지만, 렌더링의 품질 면에서는 아직까지도 OpenGL이 앞선다[7].

2.1.1.2 High Level Graphic API

고 수준 그래픽 라이브러리로, 주로 Direct3D나 OpenGL과 같은 저 수준 그래픽 라이브러리 바로 위에 존재한다. 이렇게 하면 프로그램의 이식성이 좋아지고, 복잡한 저 수준의 그래픽 함수 호출에 대한 구현이나 최적화가 필요 없고, 빠른 시간에 응용프로그램을 만들 수 있다. 여기에 해당되는 것들은 다음과 같다.

- Performer
- Open Inventor
- Vega
- Java3D
- Open Scene Graph

그리고 저 수준 그래픽 라이브러리와는 [Fig. 2-4]과 같은 구조 관계를 가진다.

Performer에 대해서 알아보기 전에 먼저 장면 그래프(Scene Graph)에 대해서 알 필요가 있다. 장면 그래프는 컴퓨터 그래픽스에서 장면을 구성하기 위해 사용되는 데이터 구조이다. 보통 장면(Scene)은 여러 가지 다른 부분들로 분해 할 수 있는데, 이러한 부분들은 서로 연결이 되어 있다. 장면 그래프는 노드 들로 구성되어 있는데, 하나의 노드는 다시 여러 개의 노드 들로 구성이 된다. 그러므로 장면 그래프란 노드 간의 계층 구조를 나타내는 그래프라고 할 수 있다. 장면 그래프를 사용하면 복잡한 3D 환경을 추상화 하여 간단하게 나타낼 수 있고, 유연성이 있으며 조작하기가 편리하다.

간단한 예를 들면 다음과 같다. 도로와 트럭으로 구성되어 있는 하나의 장면을 렌더링 한다

고 가정하자. 이것을 나타내는 장면 그래프는 [Fig. 2-5] 과 같다. 하지만, 이렇게 장면 그래프를 구성할 경우 트럭은 특정한 장소에 고정되어 있을 것이며, 내가 원하는 곳으로 움직이게 할 수 없을 것이다. 이런 경우에, 트럭을 원하는 장소로 움직이게 하기 위해서는 [Fig. 2-6] 과 같이 Translation 노드를 트럭 노드의 상위에 위치시키면 된다. 상위 노드의 속성은 모든 하위 노드로 상속이 되기 때문에 트럭은 원하는 위치로 움직일 수 있게 된다. 또 그림에서 보는 바와 같이 노드가 항상 기하 형상을 나타내는 것만은 아니다. 재질을 나타낼 수도 있고, 변환이나 물질의 속성등 을 나타낼 수도 있다. 하지만, 최상위 노드는 반드시 전체의 장면을 나타내고, 최하위 노드(Leaf Node)는 기하 형상을 나타낸다[45].

여기서 언급할 고 수준 그래픽 라이브러리들과 좀 뒤에서 설명할 VRML, X3D 은 모두 장면 그래프 구조를 사용한다.

이제 Performer에 대해서 알아보자. Performer는 OpenGL을 기반으로 하는, sgi社 의 상업용 장면 그래프(Scene graph)로써, C와 C++를 사용한 3D 그래픽스 렌더링 툴킷이다. Performer는 sgi社의 그래픽 전용 UNIX 시스템인 IRIX를 최대한 활용하기 위해 개발되었다. 또한 멀티쓰레드 기능을 지원하기 때문에, 멀티 프로세서를 사용하는 sgi社의 IRIX의 기능을 최대한 사용할 수 있다. 요즘에는 IRIX 뿐 아니라 Linux나 Windows를 지원하는 버전들도 나오고 있다. 하지만, Performer의 기능을 최대한 사용하기 위해서는 IRIX 시스템과, 전용 서버인 Onyx를 사용해야 하며, 상당히 고가이다[12].

Open Inventor는 Performer와 마찬가지로, OpenGL을 기반으로 하는 sgi社의 상업용 3D 개발 툴이다. 하지만, Performer와의 차이점은, Performer는 하드 코딩을 해야 하는 반면, Open Inventor는 사용자가 프로그래밍 실력이 부족하더라도 GUI에서 완벽히 제공하는 장면 그래프와의 상호작용으로 쉽게 3D 대상을 모델링하고 애니메이션을 수행할 수 있다[12].

Vega는 앞에서 설명한 Performer를 한 단계 더 추상화하여 만들어진 실시간 시뮬레이션 가시화 개발 툴킷이다. Vega는 Performer에 있는 모든 Scene Graph를 개발자들이 사용할 수 있게 하면서, 동시에 Lynx라는 GUI를 제공하여서 Performer에서 보다 훨씬 더 빠른 시간에 가시화 프로그램을 만들 수 있도록 한다. 그렇기 때문에, 남는 시간에 좀 더 특수화된 구현을 할 수 있다[6].

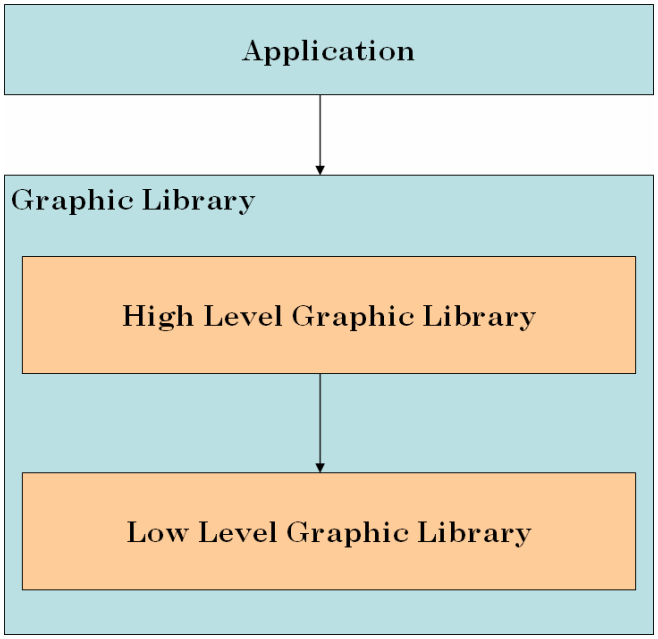

Fig. 2-4 고 수준 그래픽 라이브러리 구조[10]

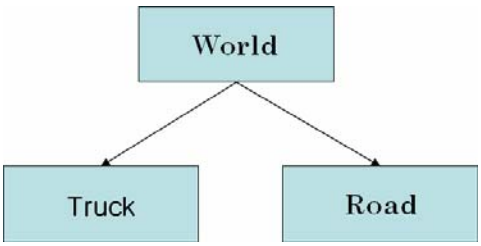

Fig. 2-5 장면 그래프, 길과 트럭으로 구성

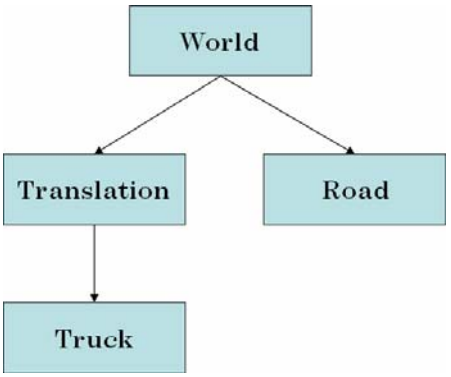

Fig. 2-6 장면 그래프, 길과 이동 가능한 트럭으로 구성

Performer, Open Inventor, 그리고 Performer의 한 단계 상위 레벨인 Vega는 플랫폼에 독립적이지 못하며, 상당히 고가라는 단점이 있다. 또한 요즘 계속 이슈가 되고 있는 웹기반의 가시화는 전혀 고려되고 있지 못하다. 이에 반해 Java3D는 이러한 단점들을 보완해 줄만한 툴킷이다. Java3D 역시 고 수준 그래픽 라이브러리로, 저 수준 그래픽 라이브러리인 OpenGL이나 Direct3D를 렌더링에 사용한다. Java3D의 장점은, 순수하게 Java만을 사용하여 프로그래밍 할 수 있다는 것이다. 아무리 큰 3D 응용프로그램에서도 랜더링 코드는 전체에서 일부분만 차지한다. 그러므로, 코드가 일관성이 있고, 이식성이 뛰어나다. 또한 이미 네트워크 컴퓨팅과 인터넷의 언어로 자리 잡고 있기 때문에, Java3D은 플랫폼 독립적이며 저가형 가시화에 적절한 가시화 방법이다[5].

하지만 단점도 존재한다. OpenGL에서는 가능하지만, Java3D에서는 불가능한 기능들이 일부 존재하며, 이것 때문에 장면이나 렌더링을 제어하지 못하는 경우가 발생할 수도 있다. 그리고, 렌더링 자체는 OpenGL과 같은 저 수준 그래픽 라이브러리를 사용하지만, 프로그램 로직등 나머지 모든 기능들은 Java를 사용하게 되므로, 성능과 속도가 현저하게 떨어짐을 알 수 있다. 그리고 중요한 문제 중 하는 바로 Java의 가장 좋은 기능 중에 하나인 Garbage Collection이다. Java 런타임, Java3D 런타임, 그리고 응용프로그램 코드는 모두 객체를 생성하게 되는데, 이런 모든 객체들은 결국 쓰레기(Garbage)가 되어서 더 이상 사용하지 않게 되며, Java 가상 머신(JVM)은 Garbage Collection을 수행하여 이러한 쓰레기(Garbage)들을 모으게 된다. 그러므로, Garbage Collection이 수행되는 동안에는 애니메이션이 중간에 멈추게 되므로 장면의 사실성이 떨어지게 된다. 마지막으로, Java3D를 웹 브라우저상에서 사용하기 위해서는 Java2 가 필요하다. 하지만, 마이크로소프트사의 인터넷 익스플로러의 JVM은 Java2 SDK를 지원하지 않으므로, 수십메가 크기의 Java2 SDK를 다운받아 설치해야 하며, CLASSPATH와 같은 설정을 손수 해 주어야 하는 불편함이 있다[5].

OSG(Open Scene Graph)는 1998년 Don Burns가 취미 삼아 IRIX에서 돌아가는 Performer 장면 그래프에 기반한 행글라이더 시뮬레이터를 그 당시에는 허술했던 Linux로 포팅하면서 시작된 프로젝트의 일환으로 개발되었다. 그 프로젝트에 함께 참여했던 Robert Osfield는 2001년에 Open Scene Graph Professional Services 라는 상업적 컨설팅과 교육을 하였고, 마찬가지로 Don Burns는 2001년 말에 Andes Computer Engineering이라는 자신의 회사를 설립하였다. 이와 함께 OSG는 대중에게 많이 알려지게 되었다.

OSG 역시 OpenGL 기반의 3D 그래픽스 개발 툴킷으로, 비행 시뮬레이터, 게임, 가상현실,

과학적 가시화에 많이 사용되고 있다. OSG는 그 출발점이 Performer였기 때문에, Performer와 비슷한 점이 많다. 사용되는 함수도 거의 비슷하다. 하지만, Performer와 다른 점은, C와 C++를 함께 사용하는 Performer에 비해 OSG는 완전히 C++만 사용하므로 완전한 객체지향적 툴이며, C++과 OpenGL가 실행되는 시스템은 어디에서나 사용 가능하다. 하지만 매번 새로 컴파일을 해야 하기 때문에 Java3D와 같이 플랫폼에 독립적이라고 말할 수는 없다. 그리고 Performer와는 달리 멀티 프로세서를 지원하지 않으며, 멀티채널 또한 지원하지 않는다. 전반적으로 OSG가 Performer보다 이해하고 개발하기가 더 쉽다.

2.1.1.3 Graphic File Format

지금까지 이야기한 그래픽 라이브러리들은 모두 응용프로그램 개발의 관점에서 보았을 때의 그래픽 툴들이다. 하지만, VRML, X3D, SEDRIS 등은 그래픽 라이브러리가 아니라 일종의 파일 포맷이다. 파일 포맷이라는 말은 어떤 관심 대상에 대한 정보만 가질 뿐 렌더링은 할 수 없다는 것이다. 렌더링은 이미 제공되고 있는 뷰어를 통해서 하게 되며, 개발자는 오로지 대상에 대한 내용에만 신경을 쓰면 된다. VRML, X3D, SEDRIS는 모두 국제 표준이므로 일정한 스펙(specification)을 따르도록 되어 있고, 이러한 포맷을 가시화 해 주는 뷰어 역시 그 스펙에 맞도록 제작되어야 한다.

VRML은 인터넷 상에서 3D 가시화를 가능하도록 하는 국제표준이다. 1994년 Mark Pesce, Tony Parisi와 같은 사람들이 주도적으로 VRML 명세화 작업을 하였고, 이후 VRML 개발을 위한 국제 기구가 세워졌다. 1995년, sgi社의 Open Inventor 파일 포맷을 기반한 VRML1.0이 발표되었다. 하지만, 동적인 애니메이션이 불가능하였고, 함께 사용할 수 있는 프로그래밍에 대한 고려가 전혀 없어서 실시간 시뮬레이션이 불가능 하였다. 이러한 문제점을 해결하기 위해 1996년 VRML2.0이 발표되었다. VRML2.0에서는 ROUTE, SENSOR, TIME 과 같은 노드 들을 추가하여서 동적인 애니메이션이 가능하도록 하였고, SCRIPT 노드를 추가하여서 프로그래밍 언어와의 통신이 가능해 지면서 실시간 시뮬레이션에 대한 기초를 마련하였다. 이듬해인 1997년, VRML97 이 발표되면서 Java와 JavaScript를 프로그래밍 언어로 추가하였고, 동적으로 장면 그래프(Scene Graph)를 바꿀 수 있게 되었다. 즉, 실시간 시뮬레이션이 가능해 진 것이다. 그리고 이때 VRML97은 국제 표준으로 자리 잡게 되었다[4].

하지만, VRML은 각각의 노드 들을 따로 관리 할 수 없다. 즉, 개발자가 사용하지 않는 기능은 포함할 필요가 없지만, VRML은 불필요한 노드 들까지 한꺼번에 다 가지고 있기 때문에

VRML 뷰어는 불필요하게 무거운 경향이 있다. 그리고 상호 운용성과 확장성에 있어서 부족한 점이 많다[3]. 또한 의미정보를 체계적으로 구조화 하기 어려운 점이 있다[15].

이런 문제점을 해결하기 위해 Web3D 컨소시엄에서는 VRML의 확장 버전인 X3D를 제안하였다. X3D는 XML 기술을 사용하여 VRML을 표현하였다. 그렇기 때문에 VRML97과 호환이 가능하고, XML을 기반으로 하였기 때문에 의미 정보의 체계적 구조화가 가능하고, 확장성이 좋아졌고, 컴포넌트화 되었다. 그리고 MPEG4와 같은 표준들과의 상호 운용성을 제공한다. 고급 렌더링 노드의 추가로 가시화의 질도 높아졌으며, 텍스트 형식의 파일 뿐 아니라 바이너리 형식의 지원으로 파일의 용량이 작아지고 압축이 가능해 져서 웹 상의 전송과 실행속도 면에서도 빨라졌다[16].

SEDRIS는 환경 데이터를 의미와 관계를 잃어버리지 않고 명확하게 표현하여 저장하며, 서로 다른 데이터 포맷 간의 교환 수단으로 사용된다. 여기서 SEDRIS가 다루는 데이터의 범위는 [Fig. 2-7] 에서 보는 바와 같이 모든 데이터를 다룬다. SEDRIS는 1994년 시뮬레이션 훈련 조종 실행국(PEO STRI)의 합동 군사 전략 훈련가 프로그램과 미국 방위 고등 연구 계획국(DARPA)의 합성 전장 프로젝트에 의해 시작되었고, 미국 국방부 주도의 표준화에서 벗어나 민간 주도의 표준화 연구로 전환되었다. 그리고 1999년에는 국제표준으로 채택되었다.

여기서 알 수 있는 것은 SEDRIS는 일종의 정적인 데이터를 저장하는 데이터베이스라는 것이다. 데이터를 가시화 하기 위해서 SEDRIS에서 제공해 주는 가시화 툴이 있지만, 기능이 많이 부족하고, 더 이상 개발이 진행되지 않아 확장이 불가능하기 때문에 사용하기 어려운 점이 있다. 그러므로 OpenFlight, VRML, Maya 등의 포맷으로 변환을 해야 한다. 하지만, 이렇게 데이터를 변환할 경우 실시간 가시화가 어렵게 되고, SEDRIS 데이터베이스에 저장된 각종 정보들이 변환 과정에서 손실되게 된다[15].

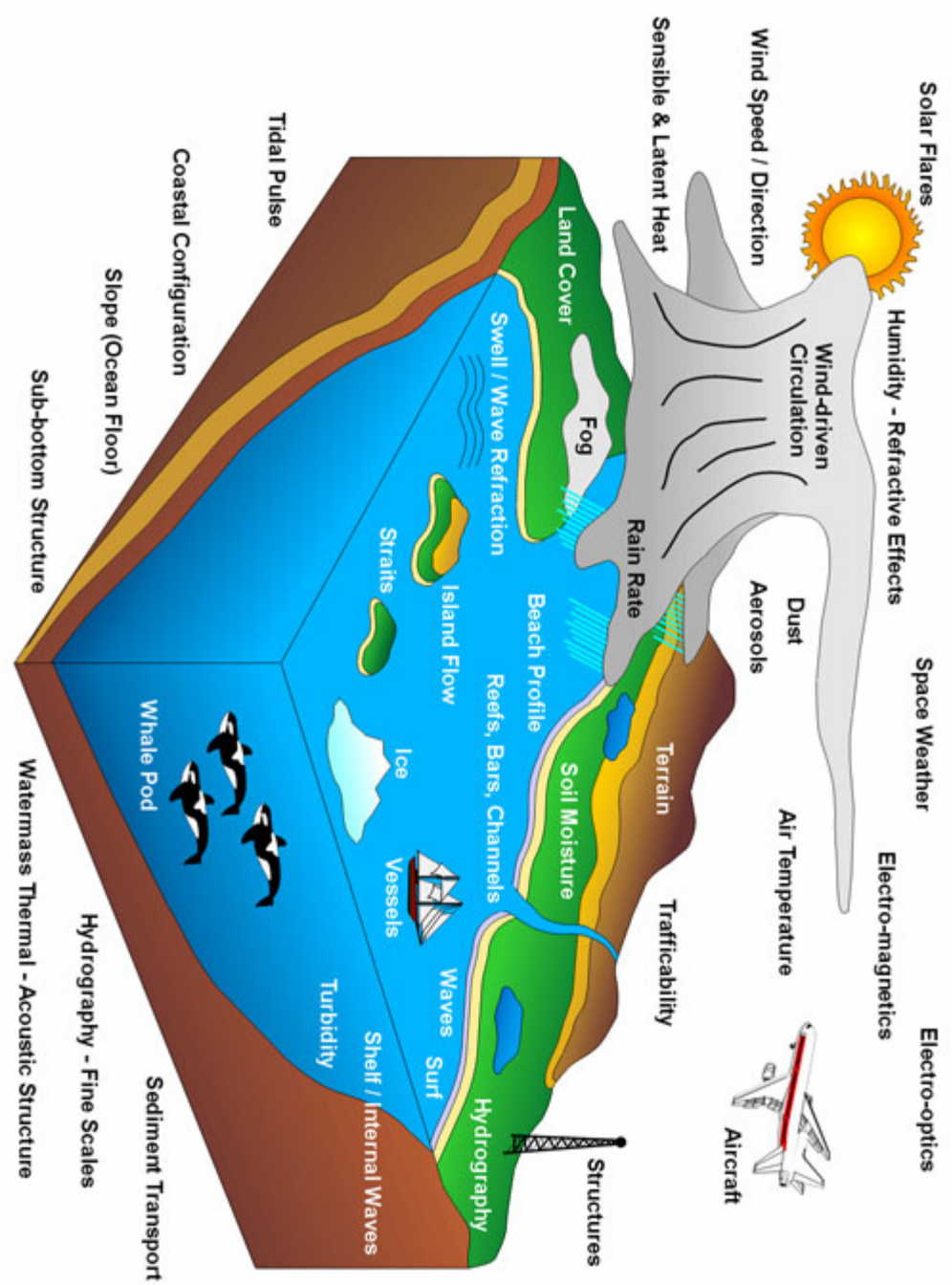

Fig. 2-7 SEDRIS에서 다루는 전체 환경 영역[52]

2.1.2. 가시화의 방법 및 포맷을 활용한 연구

앞에서 살펴 본 바와 같이 가시화를 할 수 있는 방법은 여러 가지가 있으며 각각 그 성격이 다른 것을 알 수 있다. 어느 하나의 방법이 다른 방법보다 절대적으로 좋다고 말하기 어렵고, 오히려 개발하려는 사람의 목적에 맞게 잘 선택하는 것이 더 현명하다.

여기서 각각의 방법들을 사용한 연구 사례를 살펴보고, 본 연구에서 필요한 점을 찾아 보도록 하자.

2.1.2.1 VRML

VRML을 활용한 연구가 많이 있었다.

첫번째로, Ying Li, Ken Brodlie, Nicholas Phillips가 2000년에 발표한 “Web-Based VR Training Simulator for Percutaneous Rhizotomy” [28] 라는 논문이 있다. 이 논문에서는 외과 수술 트레이닝을 위한 가상현실 기술이 너무 고가의 장비이고, 그 분야에만 사용 할 수 있다는 점을 들면서, Web 기반의 저렴하고 보편적인 시뮬레이션 방법을 제시한다. 가시화의 포맷으로 VRML을 사용하였고, 동적인 시뮬레이션을 위해서 EAI를 사용하였다. 하지만, VRML은 외과 수술에서 가장 중요한 충돌 검사를 지원해 주지 않고 있어서 따로 충돌 검사와 비슷한 효과를 낼 수 있는 다른 방법을 사용한다.

두번째로, “Multiple Window Visualization on the Web Using VRML and the EAI” [29] 에서 Jonathan C. Roberts는 하나의 시각에서가 아닌 같은 화면을 다른 시각으로 동시에 볼 수 있는 가시화 방법을 제시 한다. 가시화 포맷으로 VRML과 EAI를 사용하였다. 이 연구의 장점을 활용하기 위해 본 연구에서는 다채널 가시화를 적용한다.

세번째로, 이재철씨는 “웹 상에서 STEP 조립체 모델의 가시화”[31] 에서 3D 부품 정보를 교환하는 국제 표준 파일 포맷인 STEP형식의 데이터를 VRML과 EAI를 사용함으로서 웹 브라우저에서 조립체 모델의 데이터를 공유할 수 있음을 보였다. 하지만, VRML에 Solid에 대한 개념이 없음을 단점으로 들었다.

2.1.2.2 VEGA

Vega를 활용한 연구도 있었다. “Comparison of Vega and Java3D in a Virtual Environment Enclosure” [6] 라는 논문에서 저자인 Brian K. Christianson은 Vega와 Java3D를 이용하여 같은 모델을 가시화 하면서 두 개의 툴의 성능을 비교하였다. 그 결과 Vega는 비록 플랫폼에 종속적이고 매우 고가 이지만, Frame Rate등의 성능 면에서 월등히 좋은 결과를 보였다. 반면 Java3D는 플랫폼에 독립적이고, 무료로 사용할 수 있지만, 성능은 Vega보다 떨어짐을 보였다. 그러므로, 두 개의 소프트웨어 중에 어떠한 것이 절대적으로 좋다기 보다는 개발자가 목표로 하는 가상환경에 맞는 소프트웨어를 선택할 것을 권장한다.

2.1.2.3 Java3D

“Web-Based Simulation Visualization Using Java3D”[30] 에서 저자인 Chad F. Salisbury는 Java가 플랫폼 독립적이며, 웹 상에서도 적용이 가능한, 거의 표준과 비슷한 언어임을 강조하면서 Java3D가 Web상에서의 가시화에 있어서 정말 좋은 API임을 말한다. 하지만, Java3D를 사용하기 위해서는 반드시 Java2.0 SDK 가 필요하지만, 현재 Java2.0 SDK 을 지원해 주는 웹 브라우저가 없기 때문에 일일이 수십 메가 크기의 Java2.0 SDK를 다운로드 하여서 설치하고, Classpath를 손수 설정해 주어야 함을 이야기 하고 있다.

2.1.2.4 SEDRIS

SEDRIS를 활용한 연구로, 문홍일씨는 “SEDRIS를 이용한 디지털 생산 시뮬레이션 환경의 융합” [15] 이라는 논문을 발표했다. 그는 디지털 합성 환경 데이터와 각 사물 간의 의미정보나 관계정보를 저장하기 위해서 SEDRIS는 가장 좋은 방법이 될 수 있다고 했다. 하지만, SEDRIS에 저장되어 있는 3D 데이터를 가시화 하기에는 SEDRIS에서 제공하는 뷰어의 기능이 너무 빈약하고, 이제 더 이상 개발되지 않아 확장성이 없어서 사용하기 힘들다고 하였다. 그래서 가시화를 위해서는 SEDRIS파일인 STF 파일을 가시화 포맷인 OpenFlight나 VRML, X3D로 변환을 하는 것이 필요하다고 한다. 물론, 이렇게 변환하기 위해서는 번역기를 직접 구현해 주어야 한다. 그리고 SEDRIS의 가장 큰 장점이었던 의미정보나 관계정보가 변환과정을 거치면서 손실될 우려가 있음을 언급하였다[15].

2.1.2.5 SEDRIS

김용식씨는 “가상 공장 시뮬레이션을 위한 PC 클러스터 기반의 다채널 가시화 모듈의 설계와 구현”[11] 이라는 논문에서 가상 공장 시뮬레이션을 위해서 공개되어 있는 표준을 이용하기 보다는 특정 분야에 특화되어 있는 소프트웨어를 사용하였다. 그는 이러한 방법을 사용하면 형상정보는 물론, 의미 정보나 기구학적인 정보도 모두 저장할 수 있기 때문에 좋다고 하였다. 하지만, 이러한 소프트웨어의 가격도 고가이고, 적용 분야가 달라질 경우 기존의 소프트웨어를 사용하기 힘들고, 매번 새로운 소프트웨어를 구매해야 하는 결과를 초래한다.

2.1.2.6 적합한 가시화 방법 또는 포맷

결론적으로 앞에서 제시한 각 논문들에서 사용한 소프트웨어들은 모두 일장일단이 있다. 어느 하나의 소프트웨어가 절대적으로 좋아서 “항상 그것만 쓰면 돼” 라고 말할 수 있는 것은 이 세상에 존재하지 않는다.([Table. 2-1]참조)

본 연구에서는 플랫폼에 독립적이며 저가형의 시스템에서 시뮬레이션 결과를 가시화 하려고 한다. 플랫폼 독립적이라는 말은 그 자체로 개방된 환경이어야 하며, 누구라도 쉽게 저작하고 개발할 수 있는 환경이 되어야 한다. 그리고 저가형 시스템을 꾸미기 위해서는 S/W의 비용이 들지 않은 것이 좋다. 이러한 의미에서 볼 때 국제 표준을 사용하는 것이 가장 의미가 있다.

앞에서 컴퓨터 그래픽스를 세가지로 분류하였다. 저 수준 그래픽 라이브러리 일수록 구체적이고 자세한 표현을 할 수 있고, 성능도 뛰어나지만, 개발이 너무 어렵다. 고 수준 그래픽 라이브러리의 경우 저 수준 그래픽 라이브러리 보다 그래픽의 질이나 속도 면에서 성능은 떨어지지만, 개발의 측면에 있어서는 좀 더 쉽게 접근할 수 있다. 그렇다 하더라도 역시 손수 코딩을 해야 한다.

하지만 VRML이나 X3D와 같은 국제 표준이 파일 포맷은 응용프로그램 쪽은 신경 쓰지 않아도 된다. 오히려 컨텐츠의 개발에 집중할 수 있으며, 이것은 세가지 분류 모두 다 공통적으로 가지는 부분이므로, 국제 표준을 사용하는 것이 개발의 측면에서 보면 훨씬 더 편리하다.

플랫폼 독립적이고 저가형의 가시화 측면에서 볼 때, Java3D, VRML, X3D의 비교가 필요다. VRML은 1997년 이전에 웹 가시화를 위해서 나온 국제 표준이지만, 지금은 개발이 진행되

지 않고 있고, VRML의 단점을 보완하고 많은 기능들을 추가한 X3D가 존재하므로 VRML은 제외할 수 있다. 이제 X3D와 Java3D 중에 한가지를 선택할 수 있는데, 앞에서 분류했던 것처럼 Java3D는 High Level API로서 Application 개발의 관점의 측면으로 볼 수 있고, X3D는 Graphic File Format으로서 Content의 관점으로 볼 수 있다. 즉, X3D와 Java3D는 서로 배타적인 선택사항이 될 수 없다. 예를 들어, 본 연구에서 사용하는 Xj3D는 Java3D를 사용한 X3D 브라우저이다. 그렇다면, 둘 다 사용하는 것이 더 현명한 방법일 것이다.

한가지 생각해 볼 것이 있다. Java3D는 High Level API이기 때문에, X3D 이외에도 읽어 들일 수 있는 파일 포맷들이 존재한다. 예를 들어 VRML, 3DS, FLT, DXF와 같은 포맷들이 있다[53]. 하지만, XML을 기반으로 하고 있는 X3D가 나머지 파일 포맷에 비해 확장성이 더 좋고, 파일크기 면에서도 X3D가 더 최적화 되어 있다. 또한 웹 가시화의 측면에서 볼 때, 웹 상에서의 3D 가시화를 목적으로 만들어진 X3D가 유리한 입장에 있다.

그러므로 본 연구에서는 그래픽 파일 포맷으로 X3D를 사용하며, 브라우저로는 Java3D를 로더로 사용하는 Xj3D를 사용함으로써 저가의 플랫폼 독립적인 시스템을 구현한다.

구분	가시화 툴 및 포맷	Supporting Group	목적	플렛폼 독립성	개발 용이성	발전가능성	Open (Cost)	Web
High Level API	VEGA	Multigen-Paradigm	3D 가시화	X	△	O	X	X
	JAVA3D	Sun Microsystems		O	△	O	O	O
	OSG	www.openscenegraph.org		X	△	O	O	X
	Delmia	Dassault Systems	가상공장 가시화	X	△	O	X	X
Graphic File Format	VRML	N/A*	웹상의 3D 가시화	O	O	X	O	O
	SEDRIS	www.sedris.org	환경데이터 DB	O	△	△	O	X
	X3D	www.web3d.org	웹상의 3D 가시화	O	O	O	O	O

Table. 2-1 각 가시화 툴 및 포맷 비교표

2.2. 분산 시뮬레이션

이번 절에서는 대표적인 분산 시뮬레이션 환경에 대해서 언급하고자 한다.

2.2.1. SIMNET

SIMNET(SIMulator NETworking)은 1980년도 중 후반에 소그룹 단위의 군사 훈련을 위해 네트워크로 연결된 저가의 시뮬레이터를 만들기 위해 미국방성에 의해서 제시되었고, 개발되었다. SIMNET은 대부분의 병사들에게 훈련을 제공하기 위해 만들어 졌지만, 여전히 야전에서의 훈련을 함께 수행해야만 했다[26]. 비록 병사들의 야전 훈련을 전부 다 대체할 수는 없었지만, SIMNET은 병사들이 비싸고 위험한 전쟁 시나리오를 시험해볼 기회를 제공하였다.

SIMNET의 네트워크 구조는 분산 시뮬레이션을 하기에 매우 괜찮은 구조였다. 하지만 불행히도 그 구조가 문서화 되지 못하였다. SIMNET이 사용된 지 몇 년이 지난 후에 1991년 SIMNET의 구조를 문서화 하자는 시도가 있었고, 그 결과 1993년에 DIS 가 나왔다.

2.2.2. DIS

DIS(Distributed Interactive Simulation)는 SIMNET을 기반으로 나왔기 때문에 SIMNET과 상당히 유사한 점이 많았다. 하지만 DIS는 SIMNET 보다 더 발전된 분산 시뮬레이션이었음에 틀림 없다. DIS는 SIMNET과 마찬가지로 저가의 비용을 분산시뮬레이션을 가능하게 하였지만, SIMNET 보다 훨씬 더 많은 수의 시뮬레이션이 분산 네트워크로 연결된 가상 환경에 참여하도록 하였다. 그 결과 DIS는 큰 성공을 맛볼 수 있었다[27]. 하지만, DIS도 문제점을 가지고 있었다. DIS가 가지고 있는 패킷 정의 집합이 넓기는 하였지만, 모든 종류의 시뮬레이터들에게 적용될 만큼 보편적이지는 못하였다. DIS는 또한 새로운 형태의 정보를 나타내기 어려웠다.

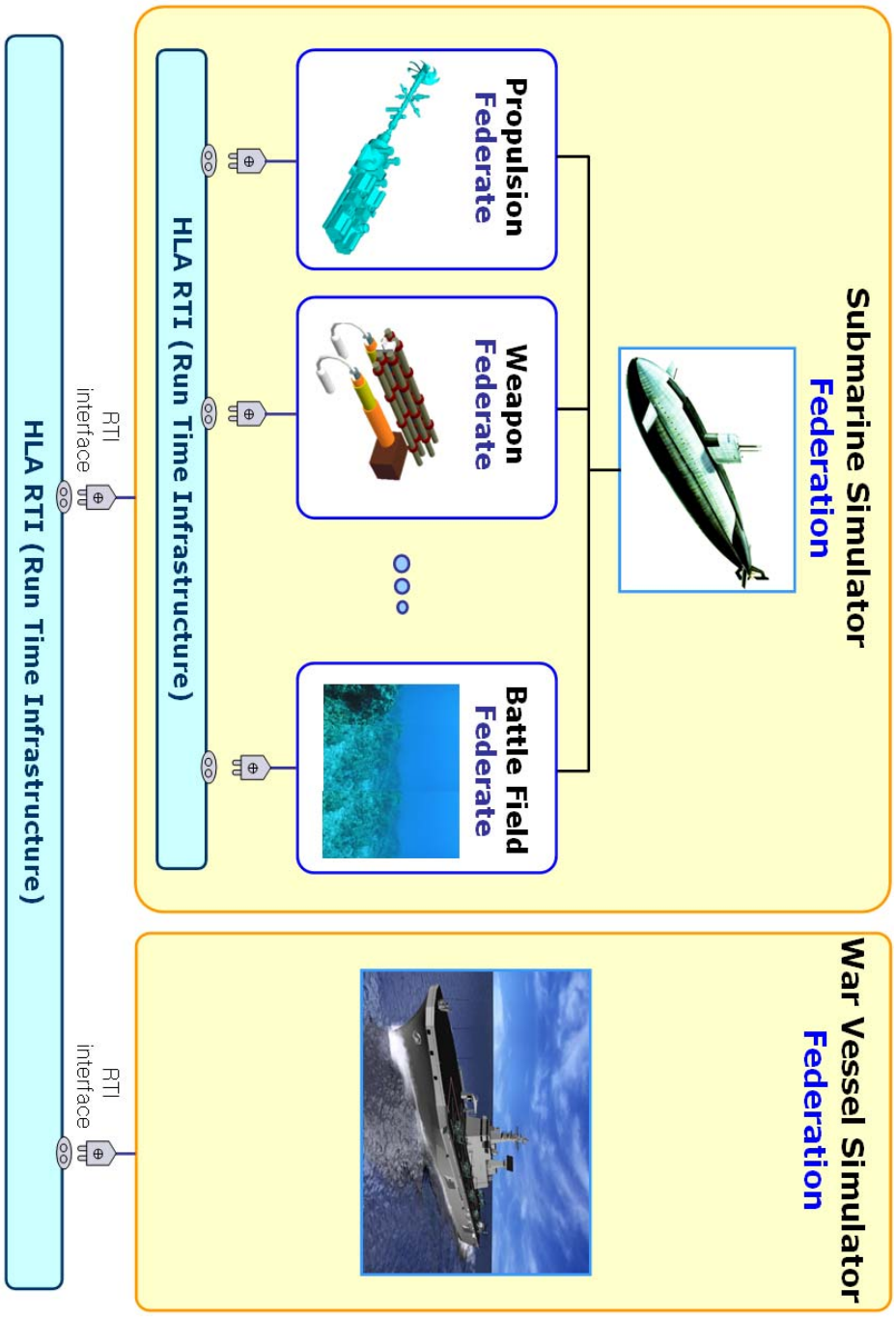

Fig. 2-8 HLA 구성도

3. X3D

요즘 3D 가속기와 같은 하드웨어가 많아졌지만, 여전히 2D 그래픽스가 오늘날의 웹 페이지를 뒤덮고 있다. 대부분의 웹 개발자들은 3D 모델들이 무거워서 다운로드 받고 렌더링 하는 것에 시간이 많이 걸리기 때문에 3D 그래픽스를 회피하려는 경향이 많다. 많은 웹 싸이트 특히 사업과 관련된 싸이트에서는 빠르게 다운로드 받을 수 있는 매크로미디어사의 플레시 에니메이션을 많이 사용한다. 그 이유는 가볍기 때문이다[14].

그럼에도 불구하고, 3D 그래픽스는 PC사양이 고성능화 되고, 네트워크의 대역폭이 넓어지면서 더 많은 잠재 가능성을 제공하고 있다.

3.1. X3D의 정의 및 특징

Web3D 컨소시엄(예전의 VRML 컨소시엄)은 최초의 웹 기반 3D 그래픽스의 국제 표준([Table. 3-1])인 VRML의 후속 버전으로서, X3D를 개발하였다. X3D는 상호작용적인 웹 기반, 그리고 브로드캐스팅 기반의 3D 컨텐츠를 정의한다. X3D는 많은 하드웨어 장치와 응용 프로그램에서 돌아가며, 통합된 3D 그래픽스와 멀티미디어를 위한 공통적인 교환 포맷을 제공한다. 새롭게 추가된 노드, 더 고급화된 API, 더 많아진 데이터 인코딩 포맷, 그리고 모듈화된 방법을 제공하는 컴포넌트 기반의 구조 또한 VRML에는 없는 X3D의 특징 들이다.

X3D가 지원하는 인코딩 포맷은 XML, VRML97, 바이너리 포맷 등 이다. 특히 XML 인코딩은 X3D에 많은 장점을 가져다 준다. XML 인코딩은 선언적인 요소와 절차적인 요소를 가지는데, 선언적인 요소를 사용함으로써, X3D는 3D, 2D 그래픽스, 애니메이션, 공간화 된 음향, 비디오 등을 포함하는 계층 구조의 장면 그래프를 사용할 수 있도록 해 준다. 그리고 절차적 요소는 스크립팅 언어를 사용하여 장면 그래프를 동적으로 변화시킬 수 있도록 해준다. X3D의 풍부한 API 는 사용자 상호작용, 네이게이션, 그리고 물리적 시뮬레이션을 제공해 준다. 그리고 어떠한 써드 파티라도 자신들 만의 X3D를 정의하고 구현할 수 있다[14].

모듈화된 구조는 X3D가 컴포넌트화 되도록 하였다. 즉, 비슷한 기능을 가지는 노드들은 같은 컴포넌트로 묶어서 정의하여 필요한 기능들이 발생할 때 추가로 확장하는 것이 쉬워졌다. 그리고 프로파일을 사용한 것이 큰 특징이라고 할 수 있다. 개발자는 자신에게 맞는 프로파일을 선택하여 컨텐츠를 개발할 수 있게 되었다. 자신에게 필요한 프로파일을 사용하는 것은

ISO-IEC 19774		Humanoid Animation
ISO-IEC 19775	Part1	X3D
	Part2	X3D API(SAI)
ISO-IEC 19776	Part1	X3D Encoding
	Part2	X3D Encoding – Classical VRML
ISO-IEC 19777	Part1	X3D Bindings - ECMAScript
	Part2	X3D Bindings – Java

Table. 3-1 X3D 국제 표준 문서[17]

모든 기능들을 한꺼번에 다 가지고 있는 VRML 보다 X3D를 더 가볍게 만든다. [Fig. 3-1]은 X3D가 가지고 있는 기본 프로파일을 나타낸 것이다.

3.2. X3D 활용분야

Web3D 컨소시엄에서 말하고 있는 X3D의 활용 분야는 다음과 같다[17].

- 공학 및 과학적 가시화
- CAD
- 건축
- 의료 가시화
- 훈련과 시뮬레이션
- 멀티미디어
- 엔터테인먼트
- 교육
- 기타

[Fig. 3-2]는 X3D 활용 분야를 그림으로 보여주고 있다.

3.3. X3D 관련 툴

앞에서도 말했듯이, X3D는 3D 장면 그래프를 가지고 있는 하나의 파일 포맷이기 때문에, X3D만을 가지고는 어떠한 것도 할 수 없다. 즉, X3D를 보여주는 뷰어가 필요하다. X3D는 국제 표준이기 때문에 여러 업체들이 이 표준에 따라 X3D 뷰어를 제작하여 내어 놓고 있다. 대표적인 뷰어로는 다음과 같다.

- Contact (Blaxxun社)[35]
- CosmoPlayer (Nexternet社)[36]
- Octaga (Octaga社)[18]
- Flux (Media Machines社)[37]
- Xj3D (Yumetech社)[38]

X3D를 저작하기 위해서는 텍스트 에디터를 이용해 손수 코딩하는 방법도 있지만, GUI를

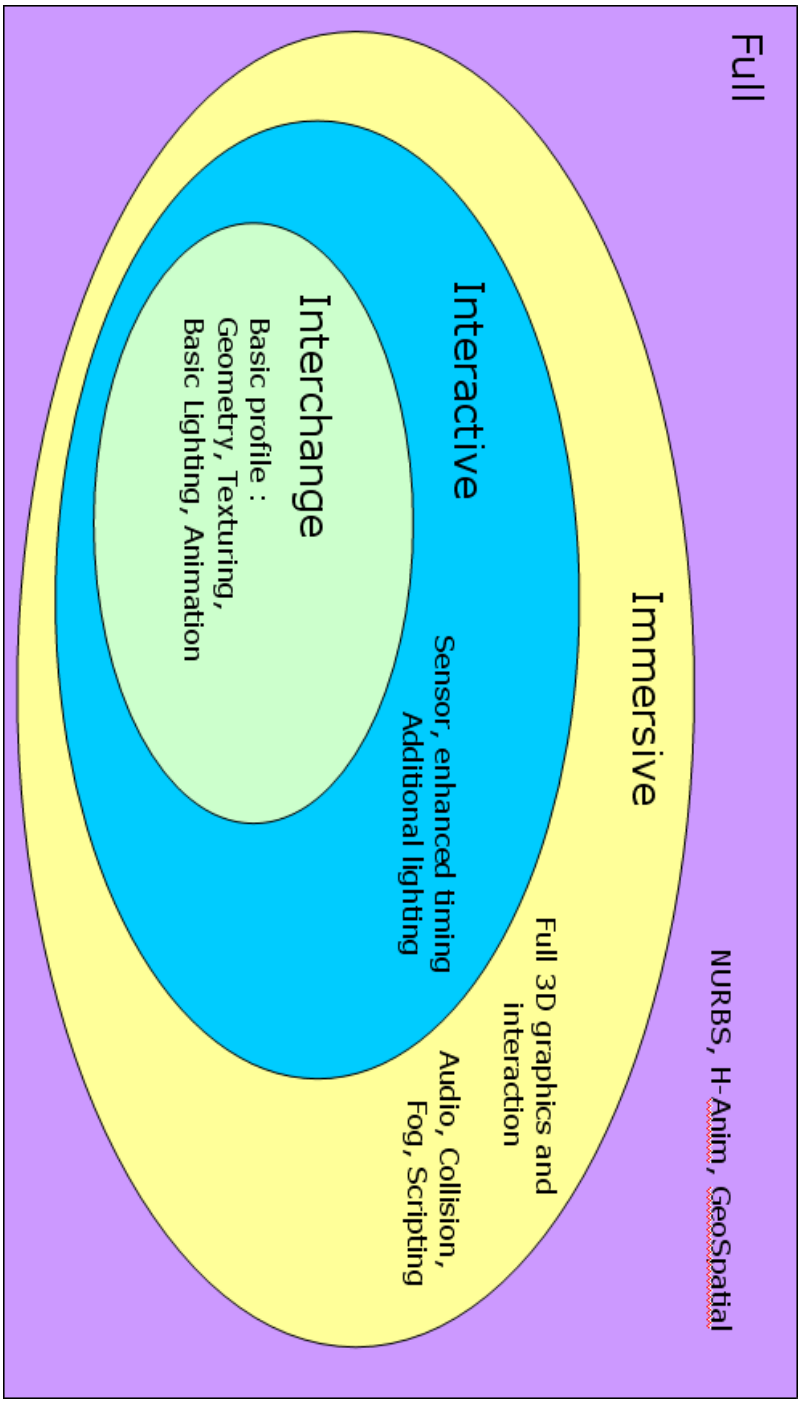

Fig. 3-1 X3D의 기본 프로파일

Fig. 3-2 X3D의 활용 분야[18]

제공하는 많은 저작도구들이 존재한다.

- Text Editor
- XML Editors
- X3D-Edit[17]
- X-Modeler(Commercial)
- Vizx3D(Commercial)

그리고, 직접 저작하지 않고 3D Max나 Maya와 같은 강력한 3D 저작도구를 이용해서 3D 모델을 만든 다음 X3D 파일로 번역하는 방법도 있다.

- Vrml97ToX3dNist[39] - NIST에서 개발한 Open Source, Free, VRML97을 X3D로 변환
- 3dsMax Exporter – Free, 3ds Max 을 X3D로 변환

3.4. X3D 활용 연구

X3D는 현재 여러 분야에서 활용되고 연구되고 있다. 이러한 연구들을 소개하면 다음과 같다.

- 교육과 실험을 위한 웹 기반 3D 그래픽스[19]
- 멘토링과 롤플레잉을 위한 재사용 가능 가상 인간[20]
- X3D 가상환경에서의 확장 가능한 상호작용[21]
- X3D를 이용한 3차원 Humanoid 모델링과 애니메이션 기법에 관한 연구[22]

3.4.1. 교육과 실험을 위한 웹 기반 3D 그래픽스[19]

이 연구에서는 군사 교육과 실험을 위해서 사용될 Web3D의 저작, 가시화, 시뮬레이션에 관한 예를 보여준다. 사용한 포맷은 X3D 이다. 논문의 곳 곳에서 X3D에 대한 자세한 소개와 저작 툴에 대한 소개가 나와 있다. [Fig. 3-3]는 이 논문에서 X3D를 사용하여 가시화한 결과를 보여 주고 있다.

Fig. 3-3 X3D를 사용한 가시화[19]

3.4.2. 멘토링과 롤플레잉을 위한 재사용 가능한 가상 인간[20]

이 연구에서는 가상 공간에서 교육을 받는 사람에게 가상의 멘토 기능을 제공하여 교육의 질을 높이려 하는 연구이다. 그리고 사람과 같은 모델링하기 어려운 대상에 대한 재 사용성에 관한 연구이다.

인간 모델은 정확도가 중요하면서도 복잡하기 때문에 표준화하기 어려운 것으로 여겨지고 있었다. Web3D 컨소시엄 에서는 가상환경에서의 인간 모델 표준으로 H-Anim(ISO/IEC 19774)를 제정하고 있다[23].

H-Anim는 X3D에서 지원하고 있는 하나의 컴포넌트이다. 이 연구는 X3D의 재사용성에 대한 검증과, 좁은 대역폭을 가지는 네트워크에서의 X3D의 사용 가능성([Fig. 3-4])을 보였다.

3.4.3. 기타 국내 논문

“X3D 가상환경에서의 확장 가능한 상호작용”[21] 의 논문에서는 X3D의 노드가 지원하는 이상의 상호 작용 방법을 제시하였고, “X3D를 이용한 3차원 Humanoid 모델링과 애니메이션 기법에 관한 연구 ”[22] 에서는 X3D의 하나의 컴포넌트인 H-Anim을 활용하여 Humanoid를 모델링한 후 그것에 애니메이션을 적용하는 방법을 제시하였다.

Recorded Video: 250KB/sec, Linear

Simulation: 2KB/sec, Interactive

Fig. 3-4 비디오와 상호작용적 시뮬레이션에 대한 비교

4. HLA

4.1. HLA 개요

DIS의 장점에도 불구하고 보편적이지 못한 단점 때문에 미 국방성 산하 기관인 DMSO(Defense Modeling and Simulation Office)는 서로 다른 목적을 위하여 개발된 여러 시뮬레이션간의 상호운용(Interoperability)과 시뮬레이션 컴포넌트의 재사용(Resue)이 가능한 HLA를 제안하였다[32]. HLA는 분산 시뮬레이션 개발의 효율 향상을 위한 객체 기반 분산 시뮬레이션 소프트웨어 구조라고 할 수 있다. HLA에서는 개개의 시뮬레이션 시스템이 상호 연동되고 시스템 구성 요소들이 재사용될 수 있도록 개념적인 정의를 내리고 있다[33]. HLA는 넓은 분산 환경 하의 다양한 시뮬레이션들이 상호간에 강력한 호환성을 가지고 긴밀한 협조 하에 동작하고, 한번 운용된 시뮬레이션들을 미래에 보다 발전된 기술과 함께 보다 편리하게 재사용하기 위한 기본 framework이다.

[Fig. 2-8]는 HLA 기반의 시뮬레이션에 대한 전반적인 개념을 담고 있다. 여기서 연동하고자 하는 각각의 시뮬레이션을 Federate이라 하며, 이러한 Federate들이 RTI라는 하부 구조를 통해 정보를 주고받는 과정을 통해 전체적인 통합 시뮬레이션이 이루어지는데, 이러한 통합적인 시뮬레이션 개념을 Federation이라 한다.

HLA의 개념이 도입된 이후, 미국 국방성에서는 모든 국방관련 시뮬레이션 개발에 HLA표준을 준수하도록 규정하고 있다. 예를 들어 차세대 전력분석용 모델인 JWAR(Joint Warfare System), 육해공군의 연합작전 모의실험 모델인 JSIM(Joint Simulation System), 무기체계 모델획득, 연구개발, 시험평가를 위한 모의실험 환경인 JMASS(Joint Modeling and Simulation System)는 모두 HLA를 기반으로 개발되고 있다[34].

2001년 이후 미국방성은 모든 국방분야 시뮬레이션들이 HLA를 따르도록 방침을 정하고 있으며, 현재 SIMNET, DIS를 따르는 Application을 HLA의 양식으로 변화시켜주는 변환에 관한 연구도 많이 진행되고 있다. 그러므로 본 연구에서는 분산환경 시뮬레이션을 위해서 HLA를 사용 한다.

4.2. HLA 용어 정리[24]

HLA와 관련된 몇가지 용어가 있다.

1) Federate : 분산 시뮬레이션을 구성하는 각각의 시뮬레이션을 말한다.
2) Federation : 여러 개의 시뮬레이션으로 구성된 분산 시뮬레이션 전체를 말하며, RTI, FOM, 그리고 여러 개의 Federate으로 구성된다.
3) RTI : HLA 기반의 분산 시뮬레이션을 동작시키기 위하여 필요한 미들웨어
4) FOM : Federation에 속한 Federate들 간에 공유해야 하는 자료의 집합
5) Federation Execution : 전체의 Federation이 실행되는 세션. 즉 Federation을 실행시키기 위해서는 Federation Execution을 생성해야 하고 Federation의 구성원이 되는 Federate들이 Federation Execution에 참여하여야 한다.

4.3. HLA 의 구성요소[24]

HLA는 Interface specification, Object Model Template(OMT), HLA rules의 세가지 specification으로 구성된다.

4.3.1. Interface specification

Interface specification은 federate와 RTI 간의 정보 교환을 위한 인터페이스 명세이다. RTI는 이러한 Interface specification에 따르는 소프트웨어이며, 전체 federation의 실행과 federate 간 정보 교환 등의 기능을 담당한다. 즉, 모든 federate 간의 정보 교환은 반드시 RTI를 거치게 된다.

4.3.2. OMT(Object Model Template)

Object Model Template(OMT)는 FOM 이 갖추어야 할 자료구조의 형식과 틀을 정의한다. 다시 말해서 OMT는 여러 가지 형태의 테이블을 정의하고 있으며, FOM은 이 테이블을 기준으로 작성된다.

FOM(Federation Object Model)은 Federation, 즉 전체 분산 시뮬레이션이 동작할 때에 Federation의 구성원인 Federate들 사이에 공유되어야 하는 실제 정보의 자료구조를 계층구조로 정의한 것이다. 예를 들어, 잠수함 Federate와 어뢰 Federate로 구성된 Federation이 있다고 가정하면, 잠수함 Federate와 어뢰 Federate간에는 서로 객체의 위치 정보를 교

환하여야 한다. 잠수함 객체, 어뢰 객체, 그리고 각 객체의 위치 속성 등이 바로 FOM에 표현된다.

4.3.3. HLA Rules

HLA Rules은 HLA 기반의 분산 시뮬레이션이 수행되기 위해서 반드시 만족시켜야 하는 기본적인 10가지 규칙들을 가리킨다. 5개는 Federation에 관한 규칙이며, 나머지 5개는 각각의 Federate가 지켜야 할 규칙으로 다음과 같다.

4.3.3.1 Federation Rules

1) Federation은 HLA OMT에 따라 작성된 HLA FOM(Federation Object Model)을 가져야 한다.
2) Federation에서 모든 Object 인스턴스는 RTI가 아닌, Federate 안에 존재한다.
3) Federation이 실행되는 동안에, Federate 간의 모든 FOM 정보 교환은 RTI를 거쳐서 전달되어야 한다.
4) Federation이 실행되는 동안에, Federate들은 HLA interface specification에 준하여 RTI와 통신하여야 한다.
5) Federation이 실행되는 동안에, Object의 Attribute의 소유권은 최대 하나의 Federate 에만 존재할 수 있다. (즉, 둘 이상의 Federate가 동일한 Object의 Attribute를 동시에 소유하지 못한다.)

4.3.3.2 Federate Rules

6) Federate는 HLA OMT에 따라 작성된 HLA SOM(Simulation Object Model)을 가져야 한다. Federation이 하나의 FOM을 갖는다면, SOM은 각각의 Federate가 한씩 가지며, 그 Federate가 제공 가능한 모든 정보의 집합을 표현한 것이다. 따라서, FOM을 구성할 때에는 Federation에 참여한 모든 Federate의 SOM에서 공통 부분을 추출하여 구성할 수 있다.
7) Federate는 SOM안에 표현한 Object Attribute를 갱신(Update) 또는 반영(Reflect)할 수 있어야 하고, Interaction을 주고 받을 수 있어야 한다.
8) Federate는 SOM 안에 표현한 Object Attribute의 소유권을 전달하거나 받을 수 있어야 한다.

9) Federate는 SOM 안에 표현된 Object Attribute를 갱신하는 모드를 전환할 수 있어야 한다.
10) Federate는 다른 Federate와 정보를 주고 받을 수 있도록 자신의 시뮬레이션 시간을 관리할 수 있어야 한다.

5. 다채널 디스플레이 및 모션 플랫폼

5.1. 다채널 동기화 개념

한 대의 PC에 하나의 화면을 가지고 디스플레이 할 경우 아무런 설정이 필요하지 않다. 하지만, 여러 대의 PC와 여러 개의 화면을 가지고 하나의 장면을 나타내야 할 경우가 있다. 흔히 가상환경 공간에서 사용자에게 몰입감을 증대시키기 위해서는 다채널의 가시화 시스템을 많이 사용한다. [Fig 5-1]을 보면 쉽게 알 수 있다. 왼쪽 그림은 하나의 화면에 잠수함 하나를 나타낸 것이고, 오른쪽 그림은 두 개의 화면에 잠수함 하나를 나타낸 것이다.

이미 고가의 상업용 시스템에서는 다채널 가시화를 지원하는 함수를 제공하기도 한다. 예를 들어, High Level Graphic API에 해당하는 Performer를 전용 컴퓨터인 Onyx에서 돌릴 경우 Multi-Processor와 Multi-Pipeline을 지원하기 때문에 간단한 설정 만으로 높은 성능을 가지는 다채널에 하나의 장면을 구현할 수 있다. 하지만, Multi-Processor와 Multi-Pipeline을 지원하지 않는 일반 PC를 사용할 경우 이러한 것은 불가능하기 때문에 다른 방법을 사용하여야 한다. 본 연구에서는 다음과 같은 간단한 알고리즘을 만들고 구현을 하였다.

같은 네트워크 상에 있는 여러 대의 PC에 동시에 같은 프로그램을 실행한 후 각각의 프로그램을 동기화 해주면 된다. 동기화를 해 주는 방법으로는 TCP나 UDP를 사용하여 동기화 패킷을 각각의 PC에 전달하는 방식을 사용한다. 동기화 해 주어야 할 PC가 많을 경우 TCP를 사용하면 속도가 느려질 수 있으므로 UDP를 주로 사용한다.

또한 여러 개의 화면으로 하나의 장면을 구현하기 위해서는 FOV(Field Of View) 계산에 의해서 Viewpoint를 적절히 구해야 한다. 구체적인 방법은 6장에서 설명한다.

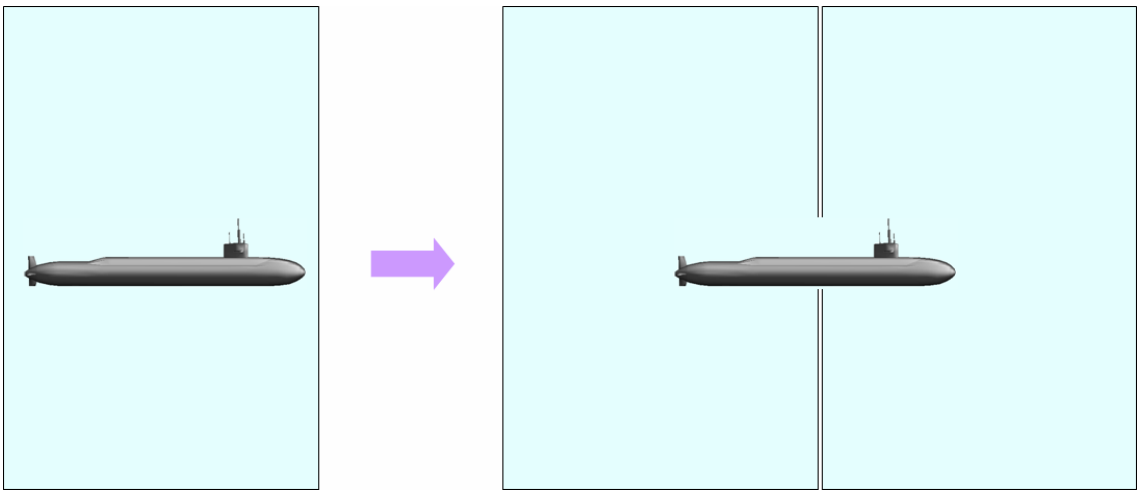

Fig. 5-1 다채널 동기화의 개념

5.2. CAVE

CAVE란 가상환경공간의 일종으로 Computer Aided Virtual Environment의 약어이다. 일반적으로 CAVE라고 말을 할 때에는 [Fig. 5-2]의 그림처럼 정육면체의 공간을 말하며, 6면에 모두 투사하여 사용자가 실제로 가상현실 공간 속에 있는 것 처럼 느끼게 한다. 각 면을 정육면체로 만든 이유는 투사면에 맺히는 이미지의 계산을 쉽게 하기 위한 것이다. 모든 빛과 색상(Hue), 대비(Contrast)가 일정하다고 가정할 경우, 사람의 시각적 예민함은 물체에 대한 직선 거리에 의존한다. 그렇기 때문에 평면의 투사면 보다는 구형의 투사면이 가장 좋다. 하지만, 컴퓨터 그래픽스는 모두 평면상의 이미지를 위해서 설계되었기 때문에 구면상의 투사는 어려운 점이 많다. 최근 들어 여러 기법을 통하여 구면 투사의 효과를 낼 수 있는 많은 연구들이 나오고 있지만, 본 연구에서는 범위를 벗어나므로 언급하지 않겠다[6].

University of Illinois at Chicago와 같은 곳에서는 회사를 세우고 CAVE를 상업적으로 팔기도 한다. 국내에서도 이미 몇몇 업체에서 이와 같은 일들을 하고 있다[46].

CAVE는 주로 문제 해결, Rapid Prototyping, 원격지로 부터의 협업 설계 등에 사용된다. 또한 자동차 산업에서 CAVE가 적용되고 있다. 실제로 사용되고 있는 예로는 General Motors, Caterpillar Corporation, NASA 등에서 사용되고 있다[6].

본 연구에서 적용할 CAVE 시스템인 iCAVE에 대해서는 6장에서 설명한다.

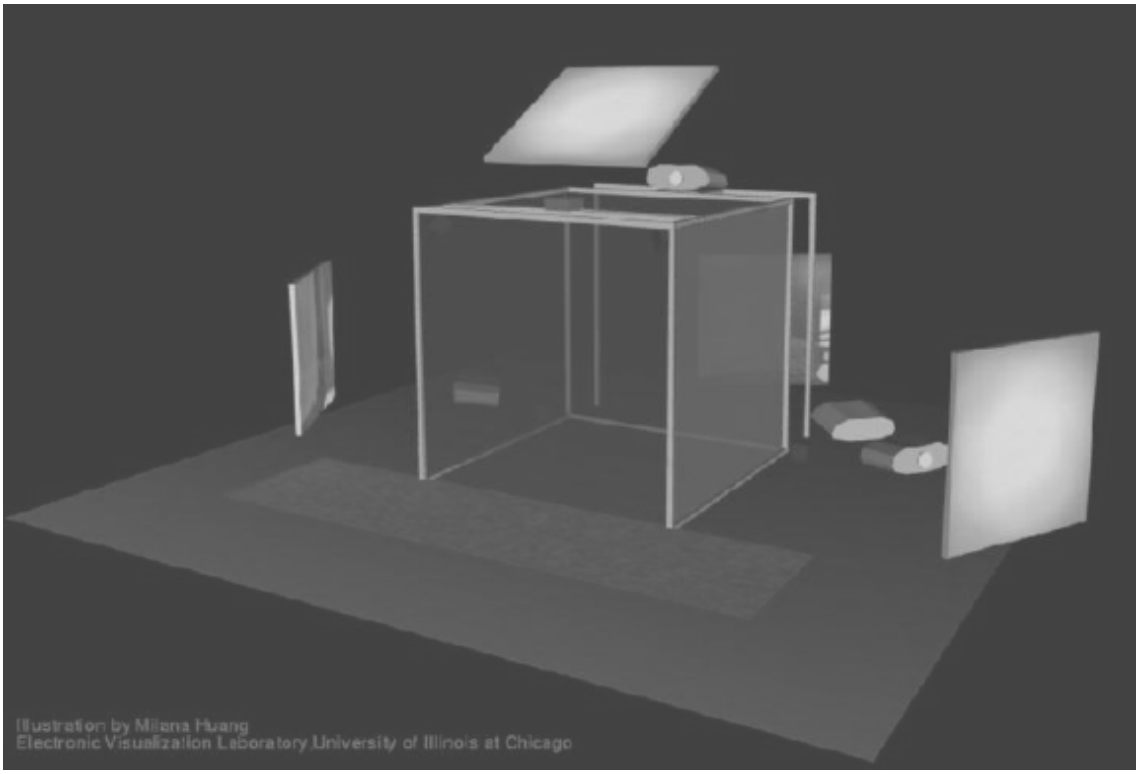

Fig. 5-2 Univ. of Illinois at Chicago에 있는 CAVE의 예

5.3. 모션 플랫폼

모션 플랫폼은 다른 말로 모션 베이스, 모션 의자, 모션 시뮬레이터 등으로 불린다. 모션 플랫폼의 주된 목적은 사용자에게 실제와 같은 운동감을 전달해 주는 것이다. 운동은 시각적 이미지와 동기화되며, 촉각이나 비디오 게임의 한 요소, 시뮬레이션, 또는 가상현실 분야에서 사용된다. 모션이 음향과 시각적 이미지와 함께 동기화 될 때 시각, 청각, 촉각의 조화를 이루게 된다. 이러한 대표적인 예가 바로 비행기 시뮬레이터 또는 자동차 시뮬레이터 이다.

가장 전형적인 모션 플랫폼은 [Fig. 5-3] 과 같은6자유도의 운동([Fig. 5-4]참고)을 갖는 Stewart 모션 플랫폼이다. 이 뿐 아니라 4자유도, 3자유도, 2자유도의 운동을 나타내는 모션 플랫폼들이 다양하게 존재한다.

모션 플랫폼이 운동감을 나타내기 위해서는 워시아웃 필터라는 것이 필요하다. 워시아웃 필터란, 예를 들어 [Fig. 5-3]에 있는 6자유도 모션 플랫폼이 어떤 운동을 표현하기 위해서 적절한 속도로 모션 플랫폼의 다리 길이를 적절하게 조절하여야 하며, 다리의 길이에 제약이 있기 때문에, 그 제약 안에서 적절한 운동감을 만들어 주어야 한다. 즉, 운동감을 다리의 길이로 나타내 주는 것을 워시아웃 필터라고 한다.

본 연구에서 사용한 모션 플랫폼과 그에 해당하는 워시아웃 필터는 뒤에서 다시 설명한다.

Fig. 5-3 KAIST에 있는 6자유도 Stewart 모션 플랫폼

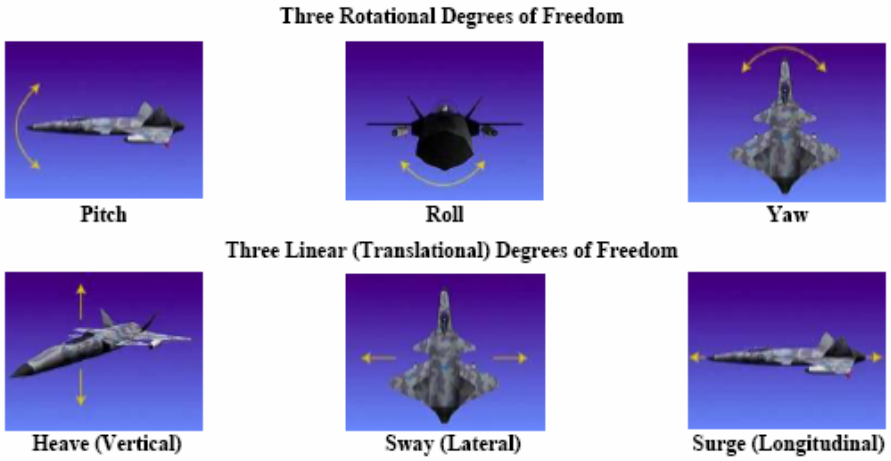

Fig. 5-4 6자유도 운동의 명칭

6. 구현 및 실험 결과

6.1. 전체 시스템

본 연구에서는 수중운동체를 공개 국제표준인 X3D를 사용하고, 저가의 모션 플랫폼과 다채널 가시화 시스템을 적용하며, 분산 시뮬레이션 환경으로 HLA를 적용한 시스템을 구현하고자 한다. 내용은 [Fig. 6-1]에 잘 나타나 있다. [Fig 6-2]는 전체 시스템이 어떻게 흘러가는가를 잘 보여주고 있다.

전체 구현 시스템을 활용할 시나리오를 생각해 보자. 부산항 앞 바다에 잠수함 두 척이 있다. 한 척은 적군이고, 다른 한 척은 아군이다. 아군은 적군 잠수함의 위치를 파악한 후, 적이 어뢰의 사정 거리 안에 들어올 때까지 적군의 잠수함에 접근한다. 접근 후, 어뢰 발사 버튼을 누르면 어뢰가 발사되고, 어뢰가 폭발하게 되면 화면이 떨리고 모션 체어에서 진동을 발생한다.

먼저, 시스템을 기능별로 간단히 분류를 하면 다음과 같이 4가지 부분으로 나눌 수 있다.

1) 입력 모듈
2) 시뮬레이션 모듈
3) 가시화 모듈
4) 모션 체어 모듈

각 기능별 모듈에 대한 구체적인 설명은 6.2 절에서 하였다.

6.1.1. 분산 시뮬레이션을 위한 federate 구성

다음으로는 분산 시뮬레이션을 위한 federate들을 구성하는 작업을 수행한다. 본 연구에서는 분산 시뮬레이션을 최대한 간단히 하기 위해서 다음과 같이 3 개의 주요 federate으로 구성하였다.

1) 입력 및 시뮬레이션 모듈 체계

사용자가 입력 모듈을 통해서 잠수함의 위치 이동을 명령하고, 그 명령에 의해서 잠수함의 위치와 자세를 실시간으로 계산한다. 예를 들어 잠수함을 특정 수심까지 잠항시키기

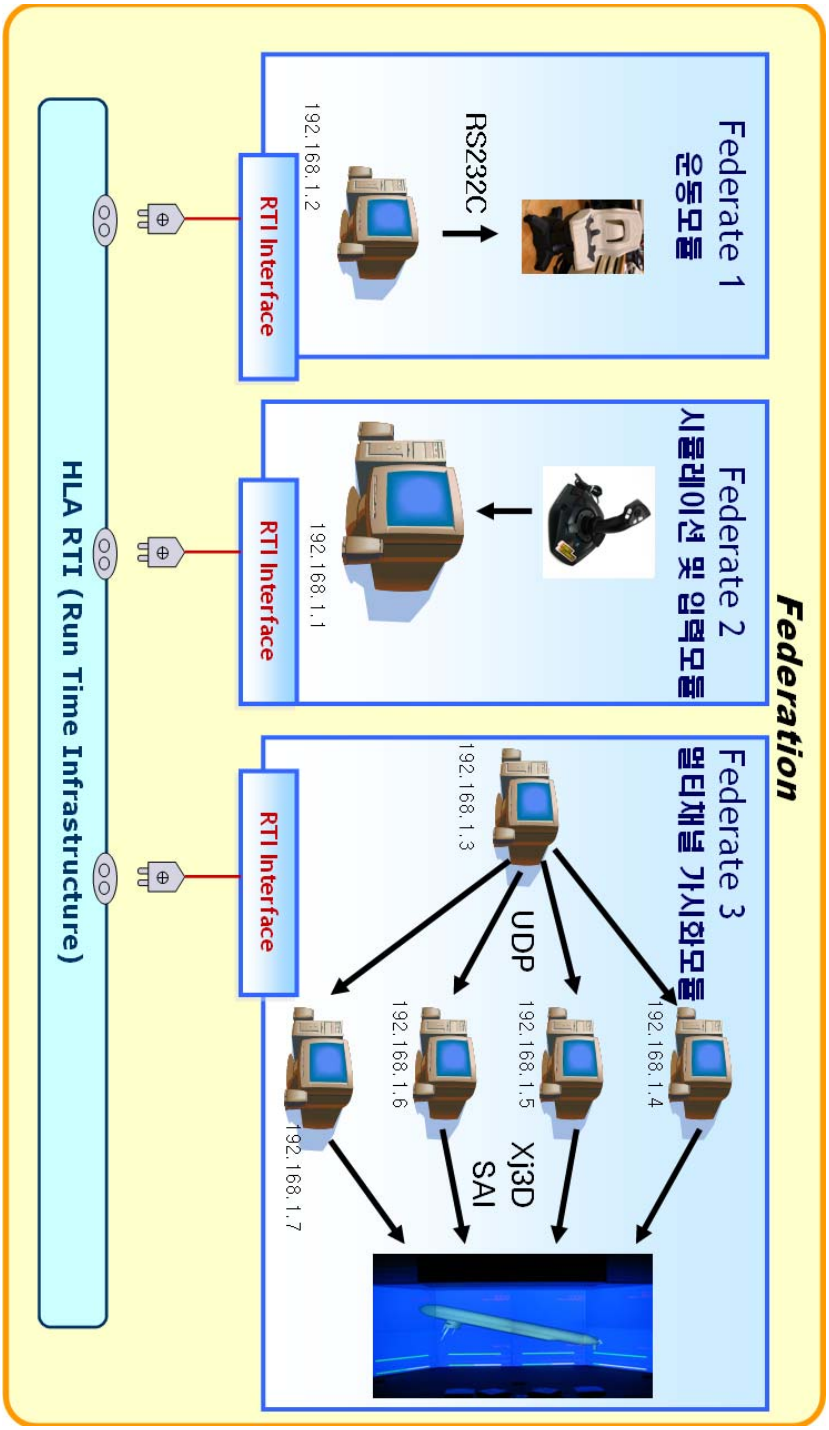

Fig. 6-1 본 연구의 구현 시스템 구조

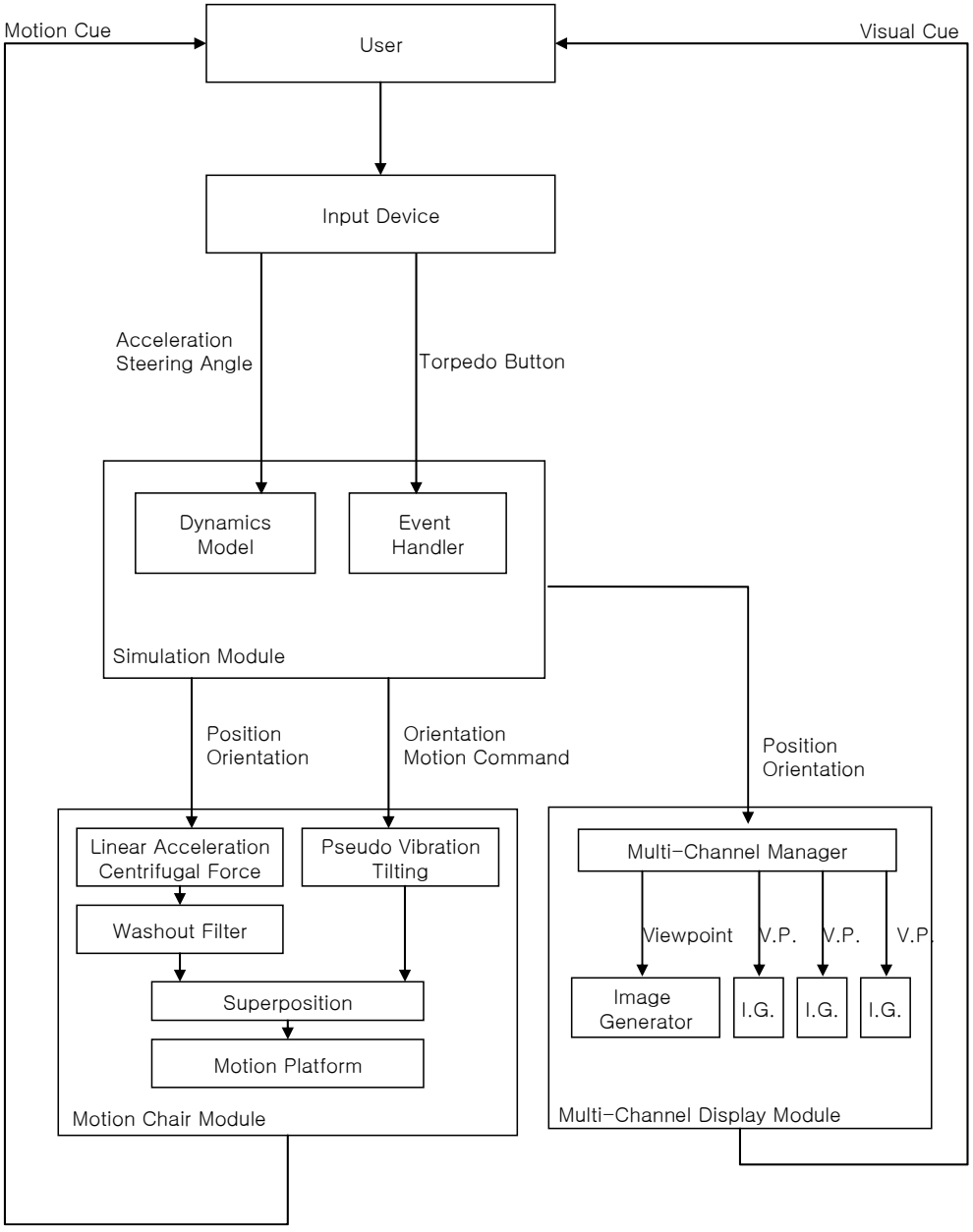

Fig. 6-2 전체 시스템 흐름도

를 원할 경우 사용자는 입력 모듈의 조작을 통하여서 잠수함이 특정 수심까지 가도록 명령을 내린다. 그리고 이 명령을 시뮬레이션 모듈이 직접 받아서 잠수함의 현재 위치와 자세를 실시간으로 계산 및 수정한다

2) 운동 모듈 체계
운동 모듈 체계는 앞의 입력 및 시뮬레이션 모듈 체계에서 계산된 잠수함의 가속도, 기울기, 진동 이벤트 등을 입력으로 받아서 모션 플랫폼을 움직여 줌으로써 운동감을 생성하는 체계이다.

3) 가시화 모듈 체계
가시화 모듈 체계는 앞의 입력 및 시뮬레이션 모듈 체계에서 계산된 잠수함의 위치 및 자세 값을 입력으로 받아서 화면에 가시화 해 주는 체계이다.

6.1.2. Federate 간에 교환해야 하는 메시지 정의

분산 시뮬레이션을 위한 federate들의 구성이 완료되면, 이들 federate들 간에 주고 받아야 할 메시지들을 정의해야 한다. [Fig 6-3]을 참조하면, 입력 및 시뮬레이션 모듈 체계가 잠수함의 현재 위치와 자세를 모두 계산해 준다. 운동 모듈 체계는 입력 및 시뮬레이션 모듈 체계에서 계산된 잠수함의 현재 자세와 모션 커맨드를 받아와서 모션 체어를 움직인다. 가시화 모듈 체계는 입력 및 시뮬레이션 모듈 체계에서 계산된 잠수함의 현재 위치와 자세 값을 받아와서 실제로 화면에 가시화 하는 역할을 한다.

[Fig 6-3]을 보면 알 수 있지만, 입력 및 시뮬레이션 모듈 체계 federate은 나머지 두 개의 federate에게 속성값을 전달해 줄 뿐 서로 상호작용을 하지는 않는다는 것을 알 수 있다. 이것은, 본 연구에서 분산 시뮬레이션의 경우 간단한 기능만을 구현하기 위해서 이와 같이 단순한 구조를 가진다.

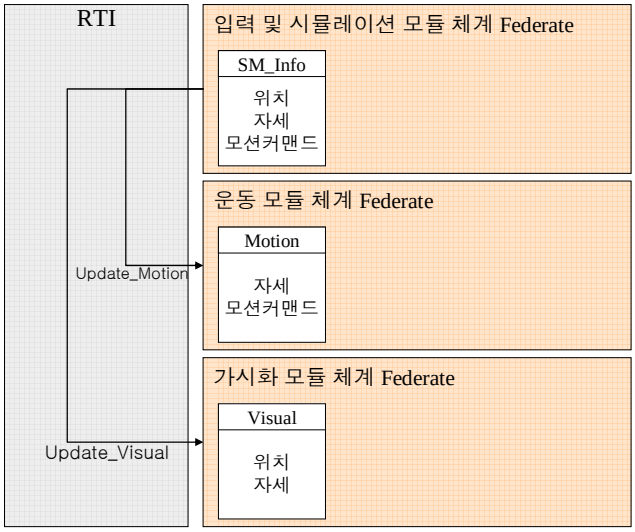

Fig. 6-3 Federate 간의 RTI를 통한 메시지 전달

6.1.3. FOM(Federation Object Model)의 작성

FOM은 모델링 단계에서 수행한 객체, 속성의 설정과 federate의 구성을 바탕으로 하여, federate들 간에 공유해야 할 객체 정보들을 추출하여 작성한다.

앞에서도 언급을 하였지만, [Fig 6-3]에서 보는 바와 같이, 본 연구에서 사용한 분산 시뮬레이션은 아주 간단하다. 3개의 federate 중 FOM에 포함되어야 할 객체는 단지 SM_Info 즉, 입력 및 시뮬레이션 모듈 체계 federate 뿐이다. 나머지 두 개의 federate 들의 Motion 인스턴스와 Visual 인스턴스는 입력 및 시뮬레이션 모듈 체계 federate 의 SM_Info 인스턴스에서 오는 정보를 받기만 할 뿐, 공개하지 않기 때문에, 자신들의 정보를 federation 에서 공유를 할 필요가 없다.

[Fig 6-4]는 FOM 중 Object 클래스와 그 속성들, 그리고 클래스 간의 상속관계를 UML(Unified Modeling Language)의 형태로 표현한 것이다. HLA의 모든 object 클래스는 ObjectRoot 클래스로부터 상속받기 때문에 모두 privilegeToDeleteObject의 속성을 갖는다. 이 속성을 가지는 federate들은 모두 객체 인스턴스를 소멸시킬 수 있다.

[Fig 6-5]는 FOM 중 Interaction 클래스의 구조를 보여준다. Update_Motion, Update_Visual 명령은 각각 자세와 모션 커맨드, 그리고 위치와 자세를 매개변수로 하는 Interaction을 전달함으로써 처리한다. SimulationEnds interaction 클래스는 시뮬레이션 종료를 명령할 때에 사용된다. SimulationEnds interaction 클래스는 Manager interaction 클래스를 상속받고 있는데, Manager interaction 클래스는 시뮬레이션 관리를 목적으로 HLA에서 기본적으로 제공하는 클래스이다.

Object Class

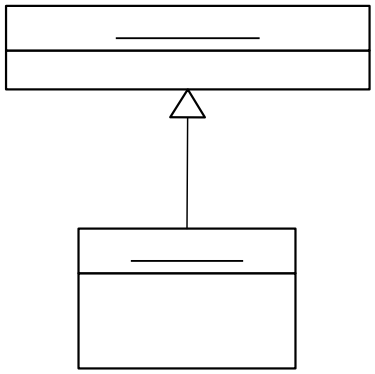

Fig. 6-4 Object Class Diagram(UML)

Interaction Class

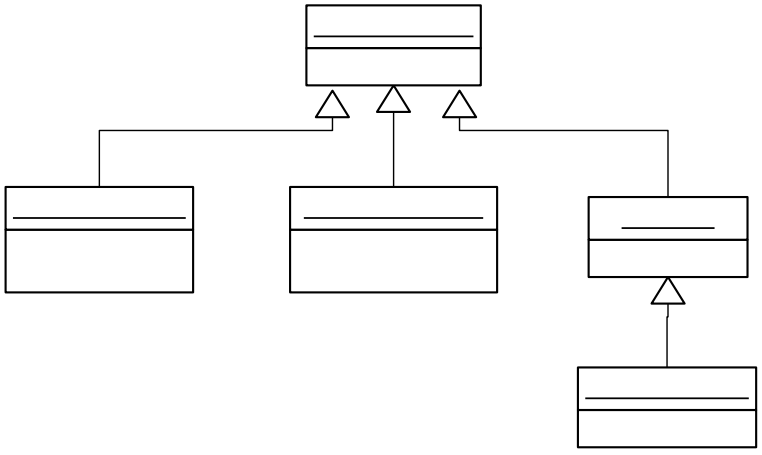

Fig. 6-5 Interaction Class Diagram(UML)

6.1.4. Performance에 관한 언급

본 연구에서는 시뮬레이션의 규모가 크지 않기 때문에 N/W 트래픽과 같은 문제는 전혀 고려대상이 되지 않는다. 하지만, 일반적으로 대규모의 시뮬레이터들을 통합하는 HLA 기반의 Application을 실행할 때에는 Performance의 상당한 저하를 가져오게 된다. 예를 들어, 초당 30 프레임의 화면을 재생성 해 주어야 자연스러운 화면을 볼 수 있지만, 대규모의 분산 시뮬레이션 환경에서는 초당 1프레임 밖에 재생성 해주지 못하는 경우도 발생한다.

이러한 환경에서도 시뮬레이션의 성능이 크게 저하되지 않게 하기 위해서 몇 가지의 방법들을 고려해 볼 수 있다.

6.1.4.1 Dead Reckoning Algorithm

분산 시뮬레이션에서 개체들 간의 정보 교환이나 개체를 제어하기 위해서 Packet을 주고 받을 경우, 특히 N/W 하부구조로서 LAN 대신 인터넷을 사용하는 경우나 대규모의 분산 시뮬레이션 환경의 경우, 제한된 N/W 대역폭에 대한 심각한 트래픽의 대역폭 요구, 혼잡으로 인한 지연시간 증가와 같은 문제가 발생될 수 있다. 이런 상황에서는 Dead Reckoning Algorithm이 좋은 해결책이 될 수 있다. Dead Reckoning algorithm이란, 개체의 위치 정보를 연속적으로 보내지 않고, 대신 timestamp, 속도 벡터를 함께 띄엄 띄엄 보내게 함으로써 추가적인 업데이트 없이 개체의 위치를 extrapolation할 수 있게 하는 것이다. [Fig 6-6]은 이러한 예를 보여 주고 있다. 가장 왼쪽의 그림과 같이 위치 정보를 최종적으로 받은 상태에서 가운데의 그림과 같이 위치 정보가 불연속적 값으로 들어올 경우, 오른쪽 그림과 같이 extrapolation을 통해서 개체의 위치를 자연스럽게 정해준다.

6.1.4.2 Multicast

Dead Reckoning Algorithm 이외에 또 다른 방법으로 Multicast를 사용하는 방법이 있다. 대부분의 분산 시뮬레이션에서 UDP 프로토콜을 사용하는데, 이 중에서도 broadcast라는 방법을 주로 사용한다. 이 방법은 같은 N/W 상에 있는 모든 호스트에 Packet을 전달하기 때문에 호스트의 수가 늘어날수록 N/W 트래픽이 증가하며, Packet을 받을 필요가 없는 호스트들도 Packet을 받게 되므로 호스트 상에도 부하가 생긴다.

최근, N/W 관련 분야에서 Broadcast 대신 Multicast라는 방법을 사용하다. Multicast 방법을 사용할 경우, 관심이 있는 호스트에 대해서만 Packet을 보낼 수 있어 N/W 트래픽을 확연히 줄일 수 있다.

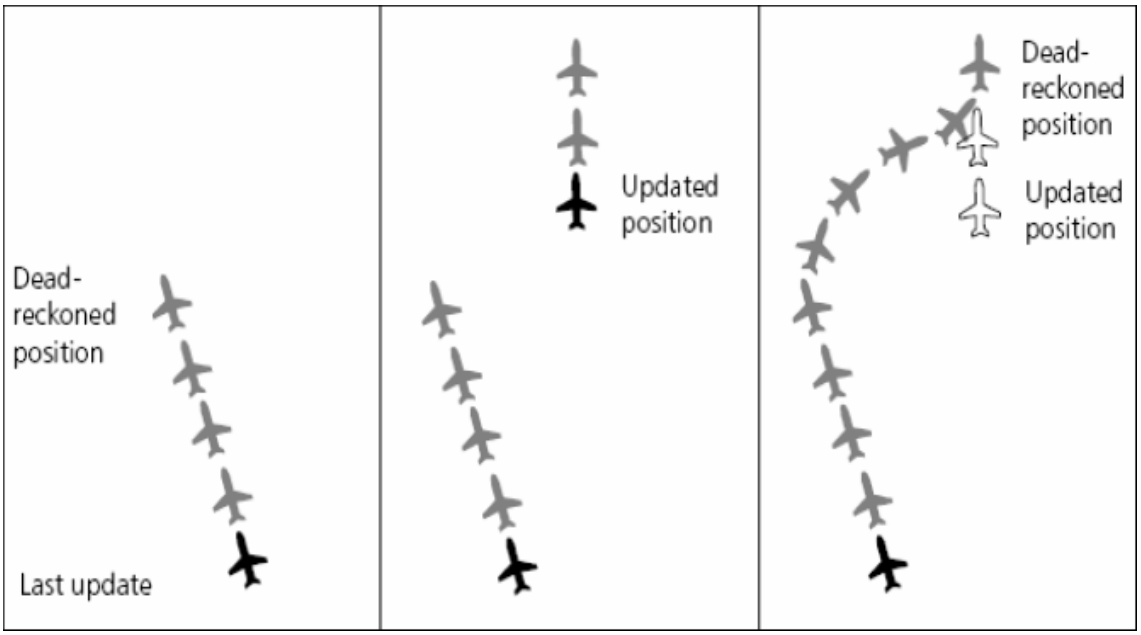

Fig. 6-6 Dead Reckoning[41]

6.2. 개별 모듈의 설계

6.2.1. 입력 모듈 설계

입력모듈은 사용자가 잠수함의 위치를 실시간으로 변화시키기 위한 것으로, 사용자가 조이스틱을 이용해서 조작한다. 구현된 시스템에서 조이스틱의 움직임에 따라 연동하여 움직이는 잠수함의 조작 부위는 다음과 같다.

1) 전후 : 잠수함 Pitch 조절
2) 좌우 : 잠수함의 Yaw 조절
3) 슬라이드 바 : 잠수함의 가감속
4) 버튼1 : 어뢰 발사

6.2.2. 시뮬레이션 모듈 설계

[Fig 6-7]는 시뮬레이션 모듈 시스템의 흐름도를 나타냈다. 이 흐름도에서 알 수 있듯이, 사용자는 모션 체어에서 얻은 Motion Cue와 가시화 모듈에서 얻은 Visual Cue를 통해서 잠수함의 새로운 위치를 입력하기 위해 입력 장치를 조작한다. 입력값으로 받는 값들은 가속도, 조향각, 어뢰 버튼 등이 있다. 가속도, 조향의 정보는 잠수함 동역학 모델에 의해서 계산되어서 잠수함의 위치와 자세를 얻는 것에 사용된다. 어뢰 발사 버튼과 같은 이벤트성 입력은 이벤트 처리기를 통하여서 모션 커맨드로 사용된다. 이러한 값을 가시화 모듈과 모션 체어 모듈에 각각 전달한다.

[Fig 6-7]의 가운데 있는 잠수함의 Dynamic 모델을 정확하게 얻을수록 정확한 결과를 얻을 수 있으나, 본 연구에서의 초점은 유체역학적 상세 모델을 구하는 것이 아니므로, 간단한 수식을 이용하여 계산한다..

이렇게 구해진 잠수함의 현재 위치와 자세 값을 HLA/RTI를 통해서 각각 가시화 모듈과 모션 플랫폼 모듈에 전달한다.

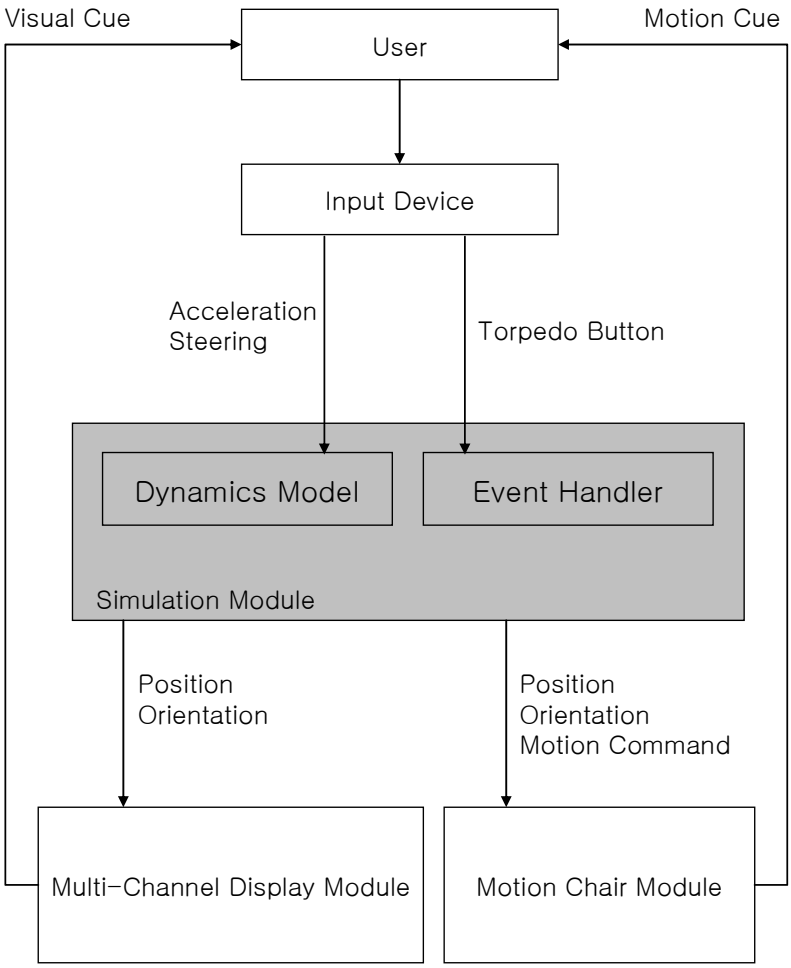

Fig. 6-7 시뮬레이션 모듈 시스템 흐름도

6.2.3. 가시화 모듈 설계

[Fig 6-8]은 가시화 모듈을 나타낸 시스템의 흐름도이다. 이 흐름도에서 알 수 있듯이, 사용자는 모션 체어에서 얻은 Motion Cue와 가시화 모듈에서 얻은 Visual Cue를 통해서 잠수함의 새로운 위치를 입력하기 위해 입력 장치를 조작한다. 입력값을 가지고 시뮬레이션 모듈은 잠수함의 새로운 위치와 자세 값을 얻게 되고, 이 값을 가시화 모듈에 전달한다. 이렇게 전달된 잠수함의 위치와 자세값은 Multi-Channel Manager에 의해서 4개의 Image Generator에 각각의 Viewpoint 정보와 함께 전달된다. 그 결과 다채널의 가시화가 가능해 진다.

기본적으로 X3D는 플랫폼 독립적이고 개방된 국제표준이므로 대부분의 일반 사양 PC 에서 가시화가 가능하다. 하지만, 문제는, 실시간으로 넘어 오는 시뮬레이션 결과값을 X3D 장면 그래프에 반영시켜서 장면 그래프를 시뮬레이션 결과값에 맞게 변형시켜야 한다.

이렇게 하기 위해서는 기본적으로 X3D 뷰어와 시뮬레이션 모듈간에 네트워크 통신이 가능해야 하며, X3D 뷰어는 X3D 장면 그래프를 동적으로 바꿀 수 있어야 한다. 이것을 그림으로 나타내면 [Fig. 6-9] 와 같다. 즉, X3D 장면은 Xj3D 엔진을 통해서 화면에 가시화 되고, Application Code는 SAI를 통해서 X3D의 장면 그래프를 실시간으로 바꾸어 주게 되는 것이다.

이번 절에서는 SAI를 통해 X3D의 장면 그래프를 실시간으로 바꾸는 것과 다채널 동기화에 대해서 언급한다.

6.2.3.1 SAI(Scene Access Interface)

X3D로 나타난 3D 모델은 Viewer를 통해서 보여지지만, 이미 X3D라는 파일 포맷에 정의된 정보만을 나타내기 때문에 X3D 장면 그래프를 바꾸지 못하여서 단순한 3D 모델만 보여주게 된다. 이런 경우 X3D의 장면 그래프를 동적으로 바꾸어 주지 못하기 때문에 M&S에서 가시화의 목적으로 X3D를 사용하기 힘들다. 이러한 문제점을 해결하기 위해서 SAI(Scene Access Interface)라는 인터페이스가 X3D와 함께 제안되었다. SAI는 X3D의 장면 그래프를 동적으로 바꿀 수 있는 인터페이스이다. 이것은 마치 VRML에서 EAI(External Authoring Interface)를 사용하는 것과 비슷하다. 그러므로 SAI를 사용하면 다양하고 친숙한 사용자 인터페이스를 구성할 수 있으며 실시간으로 X3D의 장면 그래프를 바꿀 수 있기 때문에 X3D

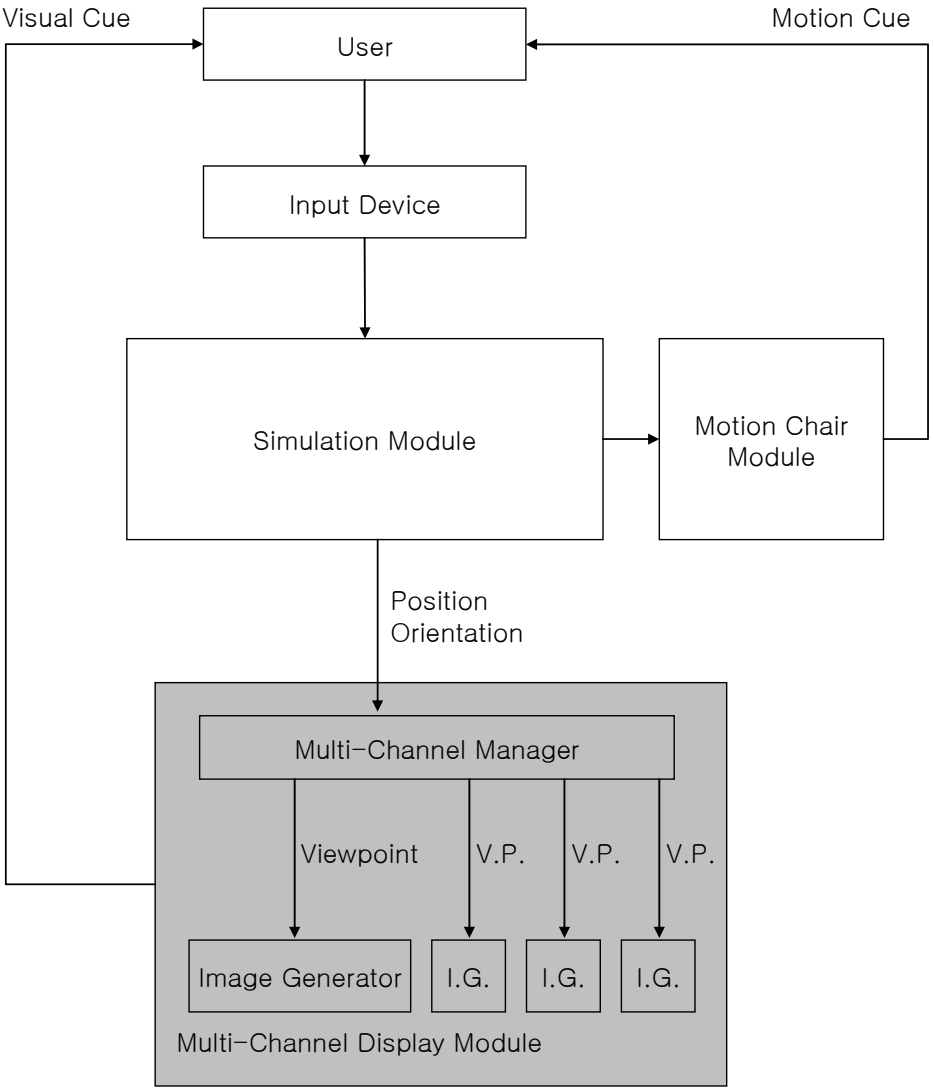

Fig. 6-8 다채널 가시화 모듈 시스템 흐름도

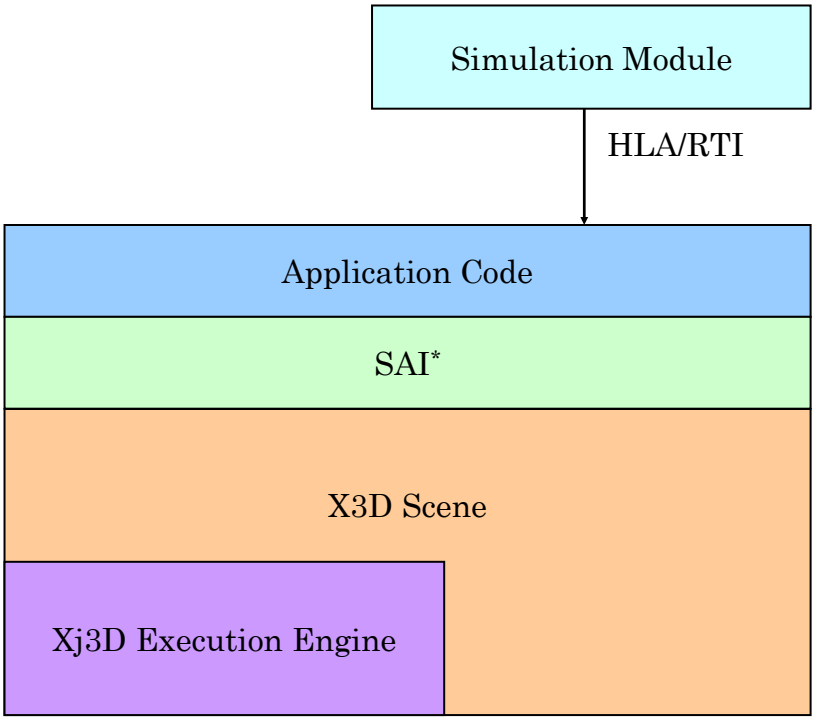

Fig. 6-9 장면 그래프를 동적으로 바꾸는 Xj3D 구조

의 M&S 가시화를 가능하게 한다[38].

SAI에 필요한 클래스들을 구현하기 위해서는 X3D Viewer 내부 구조에 대한 이해를 필요로 하기 때문에 X3D Viewer 벤더들이 해당 클래스를 번들로 제공하고 있으며 현재 사용 가능한 것으로는 본 연구에서 사용하는 Yumetech社의 Xj3D가 있다[38].

SAI가 작동을 하기 위해서는 먼저 X3D 컴포넌트를 생성해야 한다. 이는 BrowserFactory 클래스의 createX3DComponent() 메소드를 사용해서 생성할 수 있다. 그리고 이렇게 생성된 X3D 컴포넌트를 개발하려는 Application에 붙이면 된다. Pseudo 코드는 아래와 같다.

```
import org.web3d.x3d.sai.*;
public class SimpleSAI extends JFrame {
    public SimpleSAI() {
        Container contentPane = getContentPane();

        // SAI 컴포넌트를 생성한다.
        X3DComponent x3dComp = BrowserFactory.createX3DComponent(null);

        // 컴포넌트에 UI를 추가한다.
        JComponent x3dPanel = (JComponent)x3dComp.getImplementation();
        contentPane.add(x3dPanel, BorderLayout.CENTER);
```

새로운 파일을 불러오기 위해서는 브라우저에 대한 개념을 도입해야 한다. SAI에 있어서 브라우저는 시스템과 상호작용하는 인터페이스에 해당한다. 브라우저는 파일을 불러오고, 새로운 노드를 추가하고 frame rate을 얻는 등의 기능을 한다. 브라우저에 대한 참조는 Browser 클래스의 getBrowser() 메소드를 통해서 얻을 수 있다. 이렇게 브라우저에 대한 참조를 얻으면 createX3DFromURL() 메소드를 통해서 파일명을 얻은 후, replaceWorld() 메소드를 통해서 X3D 장면을 보여주게 된다. Pseudo 코드는 다음과 같다.

```
// 브라우저에 대한 핸들을 얻어온다.
ExternalBrowser x3dBrowser = x3dComp.getBrowser();

// X3D 파일을 읽어 온다.
X3DScene mainScene = x3dBrowser.createX3DFromURL(new String[] { "box.x3d" });
```

```
// 읽어온 X3D 파일을 화면에 대체하여 보여준다.
x3dBrowser.replaceWorld(mainScene);
```

하지만, 이렇게 X3D 장면을 보여주게 되면 단순히 정지해 있는 3D 물체를 보는 것 밖에 아무것도 할 수 없다. 박스를 예로 들어 보면, 박스를 움직이게 하기 위해서는 박스에 Transform 노드를 추가하고, Transform 노드의 translation 필드의 값을 계속 변화시켜주면 된다. 이것을 처리해 주는 것 또한 SAI 이다. 이것을 가능하게 하기 위해서는 먼저 Transform 노드에 "DEF"라는 정의어를 사용하여 특정한 이름을 부여한다. 이 때 getNamedNode() 메소드를 사용한다. 그 후 getField 메소드를 사용하여 translation 필드의 값을 바꾸어 주면 된다. Pseudo 코드는 다음과 같다.

```
// "MAT"라고 정의된 노드를 찾는다.
X3DNode mat = mainScene.getNamedNode("MAT");
if (mat == null) {
    System.out.println("Couldn't find material named: MAT");
    return;
}

// diffuseColor 필드를 찾는다.
SFVec3f colors = (SFVec3f) mat.getField("diffuseColor");

// 값으로 파란색을 준다.
float[] col_val = {0,0,1};
colors.setValue(col_val);
```

6.2.3.2 Xj3D[38]

이번에는 본 연구에서 사용한 X3D 툴킷인 Xj3D에 관해서 간단히 알아 본다. Xj3D는 Java 기반으로 VRML97과 X3D를 위한 툴킷을 개발하는 일종의 프로젝트이다. 많은 회사와 기관들도부터 자금의 지원을 받아서 개발을 하였다. 프로젝트의 초창기에는 Java3D API를 위한 파일 로더를 만드는 것에 초점이 맞춰져 있었지만, 지금은 X3D Specification을 잘 만족시키가를 테스트하는 것에도 많이 쓰이고 있다.

Xj3D는 일종의 X3D Application을 개발하는 툴킷이다. 자체적으로 X3D Viewer를 가지고 있으며, 많은 API들을 내부적으로 가지고 있다. 특히 앞에서 언급한 SAI를 지원하고 있어서 X3D를 이용한 M&S 의 가시화를 가능하게 한다.

Xj3D는 Web3D 컨소시엄에서 제정한 X3D Specification을 잘 따르고 있지만, 완벽하지는 않다. 특정 노드의 특정 필드를 지원하지 못하는 경우도 있고, 특정 렌더러에 따라 X3D Specification이 지원되는 비율도 다르다. 구체적인 항목에 대해서는 홈페이지를 참고하면 된다[40].

매년 크게 네 번 정도의 업데이트가 이루어 지며, 주로 Siggraph나 Web3D Conference가 열리는 시기와 일치한다. 최신 버전의 Xj3D는 2006년 4월 15일에 릴리즈 되었고, 웹 페이지를 통해서 무료로 다운로드 받을 수 있다. 물론 공개된 소스를 사용할 수 있다.

6.2.3.3 다채널 가시화 설계

5장에서 이미 언급 하였듯이, 본 연구에서는 다채널 가시화 함수를 지원하는 고가의 상업용 가시화 시스템을 사용하지 않고, 직접 다채널 동기화의 간단한 알고리즘을 개발하고 구현하는 것이 더 의미가 있는 것이므로, 본 연구에서는 직접 다채널 동기화를 구현하였다.

[Fig. 5-1]에 보면 다채널 동기화에 대한 개념을 쉽게 알 수 있다. 예로 든 그림에서는 두개의 화면에 하나의 장면을 나타낸 것이다. 이것을 구현하기 위해서는 FOV를 측정한 후에 적절한 수식을 계산하여 Viewpoint의 카메라 각을 적절히 회전시켜 주는 것이다.

[Fig. 6-10]을 보자. 사람이 화면 앞에 서서 장면을 볼 경우를 나타낸 그림이다. 가장 왼쪽의 망원경 같이 생긴 것은 화면 앞에 서있는 사람의 눈의 위치를 나타낸다. 잠수함 바로 왼쪽의 사각형은 화면을 나타내고, 잠수함 오른쪽 끝의 사각형은 물체가 보여지는 한계면을 나타낸다. 잠수함 왼쪽의 작은 사각형과 오른쪽의 큰 사각형, 그리고 두 사각형을 잇는 모서리들을 함께 보면 꼭지가 잘린 사각뿔, 즉 절두체 모양임을 알 수 있다. 이것을 Viewing Frustum 이라고 한다. 그리고 [Fig. 6-11] 에서 두 개의 Viewing Frustum을 적당한 각도로 잘 회전시켜서 Viewpoint를 설정하면 두 개의 화면으로 하나의 장면을 나타낼 수 있다.

그렇다면 어느 정도의 각으로 Viewing Frustum을 회전시켜야 할까? 먼저 [Fig. 6-12]을 하나의 화면이라고 가정하고 그림과 같이 기호를 정하자. Aspect Ratio 는 화면의 가로와 세로의 비율이므로 다음과 같이 나타낼 수 있다.

$$\begin{aligned} Aspect\ Ratio &= \frac{\text{Resolution Width}}{\text{Resolution Height}} \\ &= \frac{a}{b} = \frac{2l \times \tan\theta}{2l \times \tan\phi} \\ &= \frac{\tan\theta}{\tan\phi} \end{aligned}$$

컴퓨터의 가로, 세로 해상도는 이미 설정이 되어 있고, 스크린의 가로, 세로 길이 또한 모두 알고 있기 때문에 Aspect Ratio(화면의 가로 세로 비율)는 구할 수 있다. 그러므로 θ와 ϕ의 관계를 알 수 있다.

사용자가 화면으로부터 서있는 위치를 l이라고 하면 앞에서 구한 관계식에 의해서 θ와 ϕ를 다음과 같이 모두 구할 수 있다.

$$\begin{aligned} a &= 2l \times \tan\theta \\ \therefore \theta &= \tan^{-1}\frac{a}{2l} \\ \therefore \phi &= \tan^{-1}\frac{\tan\theta}{Aspect\ Ratio} \end{aligned}$$

FOV(Field Of View, 시야각)는 정의에 의해서 θ와 ϕ 중 작은 값의 2배임을 알 수 있다.

$$FOV = 2 \times \min(\theta, \phi)$$

이렇게 구한 FOV를 시스템에 적용시킨 후, Viewpoint를 θ 만큼 회전시켜 주면 두 개의 화면으로 하나의 장면을 보여줄 수 있다.

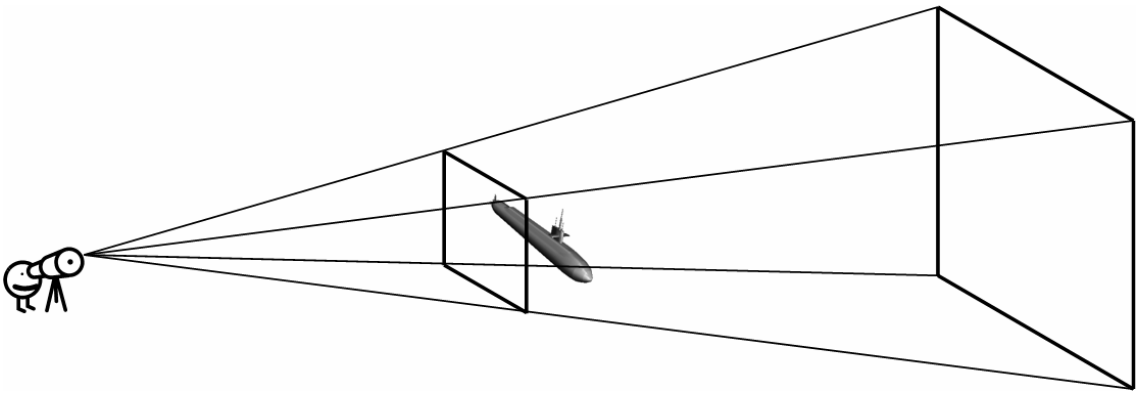

Fig. 6-10 단일 채널에서의 Viewing Frustum

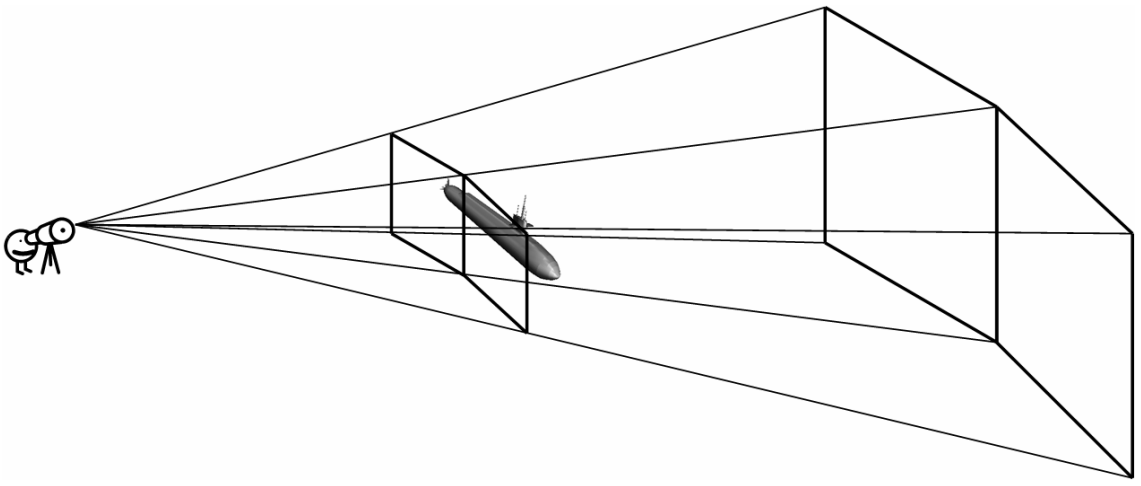

Fig. 6-11 다채널에서의 Viewing Frustum

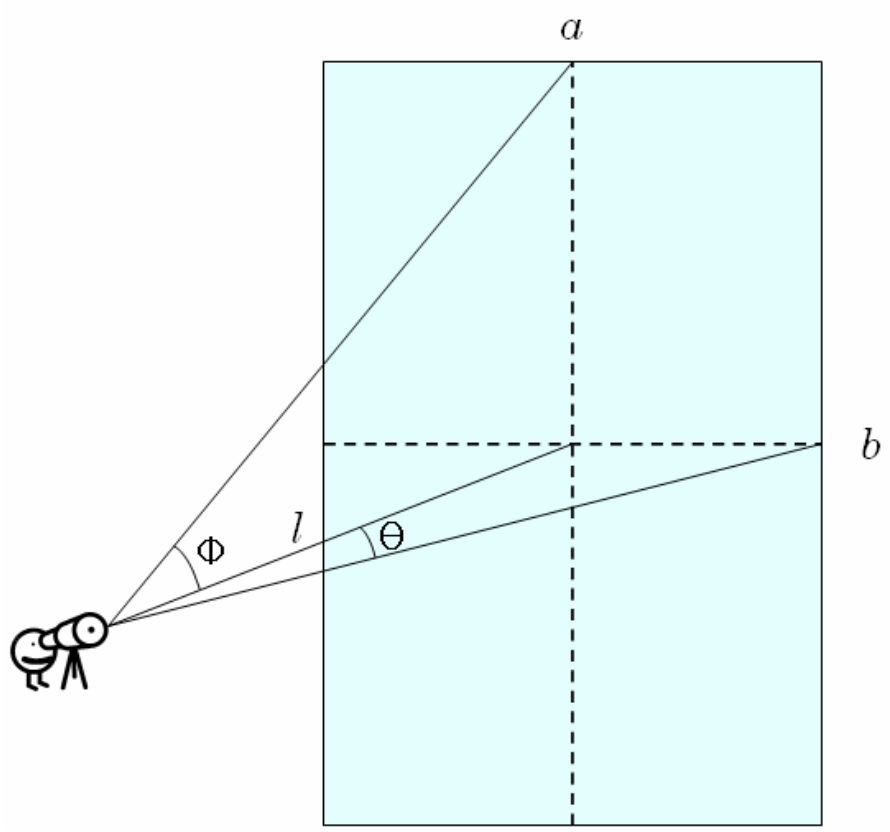

Fig. 6-12 단일채널의 측정치 변수들

6.2.3.4 iCAVE

본 연구에서 앞에서 말한 다채널 가시화 시스템을 적용할 CAVE 가상공간환경은 [Fig. 6-13]에서 보는 바와 같이 한국과학기술원 내에 구축된 가상환경공간으로, 그 이름은 iCAVE 이다.

1900x1400 mm 크기의 스크린이 모두 8개가 있고, 4대의 디스플레이용 PC, 8대의 Front Projector들이 있다. 각각의 디스플레이 PC들은 100MBps Ethernet으로 연결되어 있으며, Geforce 6800 그래픽 카드를 이용하고 있다. CPU는 Pentium 4 2.4GHz이며, Windows XP를 주 운용체계로 사용한다. 그래픽 카드 하나에 두 대의 Projector가 연결되어 있으며, LAN을 통하여 마스터 PC로부터 제어를 받아 전체가 통제된다. 또한, 각각의 PC는 1024X1536의 해상도를 가지므로, 전체 화면은 4096X1536의 해상도를 가진다.

향후 이 다채널 가시화 시스템은 Stereo의 추가, 바닥면과 천정에 대한 스크린 설치 등을 고려하고 있다.

4대의 PC는 모두 LAN을 통해서 연결이 된다. 여기서 사용할 수 있는 N/W Protocol은 TCP/IP와 UDP/IP가 있다. TCP(Transmission Control Protocol)/IP(Internet Protocol)는 연결 지향성으로서, 여러 PC들 간의 연결을 보장해 주는 N/W Protocol이다. 연결이 보장되기 때문에, N/W 상에서 전송되는 Packet의 손실이 없다. 그러나, 만약 N/W에 장애가 발생할 경우, TCP/IP는 연결의 보장을 위해서 Packet을 재전송 하는 등의 과정을 거치기 때문에 속도가 느려지는 단점이 있다. UDP(User Datagram Protocol)/IP 는 비연결 지향성으로서, 여러 PC들 간의 연결을 보장하지 않는다. 연결을 보장하지 않으므로, Packet의 손실이 발생할 수 있으며, 전송되는 순서 또한 뒤바뀔 수 있다. 하지만, Packet을 보낸 후 Packet에 대한 완전성을 신경쓰지 않기 때문에, TCP/IP에 비해 속도 면에서 빠르다는 장점을 가진다. 본 연구에서 4대의 PC를 동기화 하기 위해서 iCAVE라는 제한된 공간에서 사용하며, 4대의 PC만 같은 N/W 상에 존재하므로 외부와는 단절되어 있다. 이럴 경우, UDP/IP의 Packet 손실이나 도착 순서의 뒤바뀜 등은 고려하지 않아도 되므로, 속도에서 우위를 가지는 UDP/IP를 사용한다.

Projection Type	Front Projection
Size	1900x1400mm (for each Channel)
No. of Channels	8 Channels

Fig. 6-13 KAIST에 있는 iCAVE 가상환경 공간

6.2.4. 모션 체어 모듈 설계

[Fig 6-14]은 모션 체어 모듈 부분을 나타낸, 시스템의 흐름도이다. 이 흐름도에서 알 수 있듯이, 사용자는 모션 체어에서 얻은 Motion Cue와 가시화 모듈에서 얻은 Visual Cue를 통해서 잠수함의 새로운 위치를 입력하기 위해 입력 장치를 조작한다. 입력 값을 가지고 시뮬레이션 모듈은 잠수함의 새로운 위치와 자세 값을 얻게 되고, 이 값이 모션 체어 모듈에 전달이 된다. 값을 입력 받은 모션 체어 모듈은 선형 가속도와 원심력을 계산하고, 이것을 워시아웃 필터를 통해서 모션 체어에 적용 가능하도록 변환한다. 또한, 진동이나 기울어짐과 같은 효과를 직접 추가하여 앞의 값과 중첩(Superposition) 함으로써 운동감을 증강시킨다.

6.2.4.1 2자유도 모션 체어

본 연구에서는 운동 생성 시스템의 소규모화와 저비용화를 위해서 2자유도(Roll, Pitch)의 모션 플랫폼을 사용하였다. 하지만, 고차원의 자유도를 갖는 모션 플랫폼에 비하여 저가의 소규모 2 자유도 모션 플랫폼이 갖는 일반적인 문제점은 다음과 같다.

- Sway, Heave, Surge, Yaw 모션의 부재
- Lack of Vibration
- Rigidity problem of motion chair body
- Backlash problem

본 연구에서는 [Fig. 6-15]에서 보는 바와 같이 2자유도의 운동감(Roll, Pitch)만을 표현해주는 모션 체어를 사용한다. 이 모션 체어는 RS232C 직렬통신 프로토콜을 사용하여서 움직이며, 반드시 COM1 포트와 4800 bps 로 설정을 해 주어야 한다. 이 모션 체어를 구동하기 위한 패킷은 다음과 같다.

40 4D YY XX VY VX 00 PP YY : 7F일 때 중간, 00~FE의 값을 가지며, 앞~뒤로 움직임 XX : 7F일 때 중간, 00~FE의 값을 가지며, 좌~우로 움직임 VY : 앞~뒤의 속도를 나타냄. 00~0A의 값을 가지며, 0A일 때 가장 빠름. VX : 좌~우의 속도를 나타냄. 00~0A의 값을 가지며, 0A일 때 가장 빠름. PP : 앞의 7Byte들의 합을 나타냄 (Check Sum)

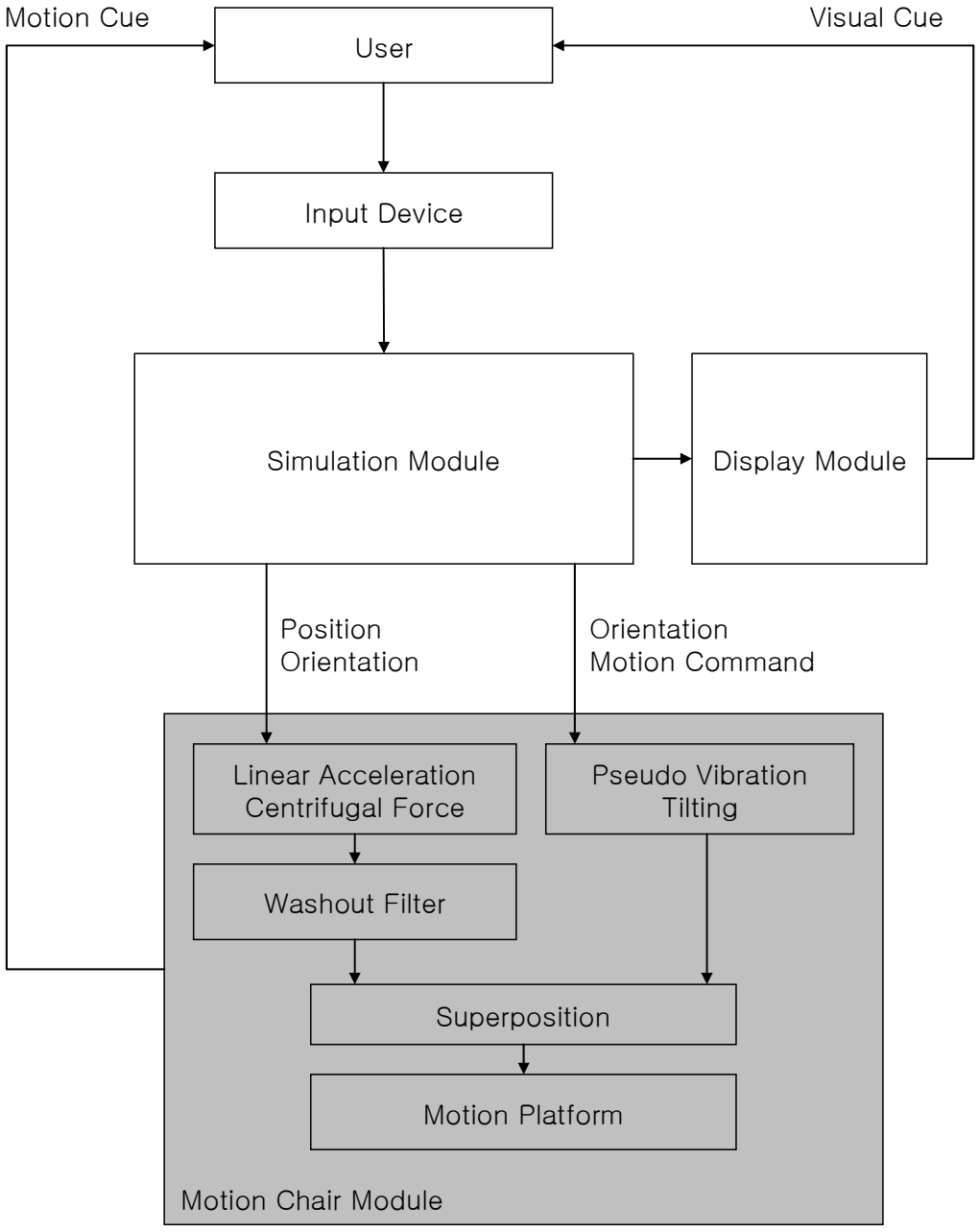

Fig. 6-14 모션 체어 모듈 시스템 흐름도

전원	220V, 60Hz
소비전력	286Wh(작동중) 176Wh(대기중)
최대 유효각	전후, 좌우 모두 7.5˚
인터페이스	직렬포트 COM1 Baud rate : 4800 bps

Fig. 6-15 2 자유도 조이체어

6.2.4.2 워시아웃 필터

2자유도의 소규모 모션 체어는 이론적으로 Sway, Heave, Surge, Yaw의 움직임을 표현하지 못하기 때문에 실제 가상현실 컨텐츠와의 연동을 위해서 이를 보상해 주기 위한 워시아웃 필터를 적용하여 자유도의 감소로 인한 현실감의 저하를 보상할 수 있다. 워시아웃 필터란, 예를 들어 [Fig 5-3]에 있는 6자유도 모션 플랫폼이 어떤 운동을 표현하기 위해서 모션 플랫폼의 다리 길이를 적절하게 조절하여야 하며, 다리의 길이에 제약이 있기 때문에, 그 제약 안에서 적절한 운동감을 만들어 주어야 한다. 즉, 운동감을 다리의 길이로 나타내 주는 것을 워시아웃 필터라고 한다.

본 연구에서 사용 할 2자유도의 모션 체어는 두 개의 액츄에이터의 회전각을 조절하여 몸체의 기울어짐을 나타낸다. 이 기울어짐을 적절하게 잘 조합하여 원하는 운동감을 나타낼 수 있는 단순한 워시아웃 필터를 만들 수 있다. 일반적으로 뉴턴의 운동법칙인

$$F=ma$$

에 의해서, 질량이 일정하다고 가정할 경우, 운동감은 가속도로 표현할 수 있다.

하지만, 문제가 되는 것이 있다. 2자유도 모션 체어는 회전운동만을 나타낼 수 있을 뿐, 선형운동은 나타내지 못하기 때문에, 실제적으로 가속도를 표현할 수 없다는 것이다. 이에 가속도와 유사한 운동감을 줄 수 있는 방법으로, 모션 체어의 병진 운동이 아닌 기울어짐을 사용할 수 있다.

선형 가속도의 경우를 생각해 보자. 자동차가 급정거를 하거나 급출발을 할 때, 타고 있는 사람은 몸이 앞이나 뒤로 쏠리는 것을 느낄 수 있다. 그러므로, 앞뒤로의 가감속을 나타내는 선형 가속도가 양의 방향으로 조금 작용한다면, 모션 체어를 뒷 방향으로 조금, 그리고 천천히 기울여 주어서 관성을 느끼게 할 수 있고, 크게 작용한다면 모션 체어를 뒷 방향으로 많이, 그리고 빠르게 기울여 주어서 큰 관성을 느끼게 할 수 있다. 즉, 선형 가속도의 크기는 앞 뒤로의 기울어짐으로 나타낸다.

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

원심력 : $m\frac{v^2}{r}$

m : 잠수함의 질량, v : 잠수함의 진행방향 속도, r : 잠수함의 회전반경

또한, 워시아웃 필터와 관계없이, 진동효과나 잠수함의 기울기를 직접 모션 체어에 전달하여 중첩시킴으로써, 운동감을 증강시킬 수 있다([Fig 6-14]참조). 진동은 Heave 모션이 있어야 가능해야 하지만, Heave 모션이 없을 경우, Roll과 Pitch 모션을 임의적으로 주어서 모션 체어가 전후 좌우로 랜덤하게 왔다 갔다 함으로써, 유사 진동을 줄 수 있다.

이렇게 구해진 선형 가속도, 원심력, 기울어짐, 진동은 모두 모션 체어의 Roll, Pitch 각으로 나타나므로, 중첩(Superposition) 시킴으로써, 최종 값이 모션 체어에 적용이 된다.

6.3. X3D 모델 획득

X3D 가시화를 위한 파일 포맷 변환 과정은 다음과 같다.

1단계 : Open Flight 파일을 VRML 파일로 변환

해저 지형, 잠수함, 어뢰 등의 X3D 파일이 이미 존재할 경우에 파일 변환이라는 작업이 필요하지 않다. 그리고 처음부터 X3D 파일 포맷으로 3D 모델링을 할 필요는 없다. 본 연구에서는 이미 Open Flight 파일 형태로 연구에 필요한 모델들이 존재하였기 때문에, 이 파일 포맷을 X3D로 변환하는 것이 바람직하다. 하지만, Open Flight 포맷을 X3D 포맷으로 바로 바꾸어 주는 변환툴이 없고 대신 VRML로 변환해 주는 툴이 존재하므로, Open Flight 파일을 VRML 파일 포맷으로 변환을 한다. 여기서 사용된 변환툴은 Right Hemisphere社[43]의 Deep Exploration CAD Edition 이다. Deep Exploration CAD Edition 은 변환툴로서 아주 좋은 성능을 가지고 있다고 알려져 있다.

2단계 : VRML 파일을 X3D 파일로 변환

1단계에서 변환된 VRML 파일을 X3D 파일로 변환을 해야 한다. 여기서 사용된 변환툴은 NIST에서 만들어 공개 배포한 Vrml97ToX3dNist 이다. 이 변환툴은 Web3D에서 발표한 x3d-3.0.dtd 에 따라 만들어 졌다.

3단계 : X3D 파일을 Xj3D가 읽을 수 있도록 추가 수정

2단계에서 만들어진 X3D 파일을 Xj3D에서 바로 읽어 들였을 때 렌더링 하지 못하는 상황이 발생하였다. 그 이유를 살펴보니 다음과 같았다. Deep Exploration CAD Edition으로 Open Flight 파일을 VRML로 변환할 때 각각의 노드를 가리키는 DEF문 중에 한가지 이름이 중복적으로 나타나는 현상이 발생하였다. 그리고 다른 한가지는, VRML 포맷을 X3D 파일로 변환을 할 때 Vrml97ToX3dNist 변환툴이 변환한 필드 값 중에 Xj3D가 인식하지 못하는 필드값이 있었다. 예를 들어 Vrml97ToX3dNist 변환툴은 프로파일 필드에 "Full"을 사용하였지만, Xj3D는 "Full" 까지 지원을 하지 못하였고, 바로 아래 단계인 "Immversive" 를 지원하였다. Xj3D가 x3d-3.0.dtd를 따르고 있지만, 몇 가지 필드들은 아직 따르고 있지 않음을 알 수 있다[40]. 그래서 Text Editor을 이용하여 Xj3D가 읽을 수 있도록 위의 오류들을 수정해 주었다.

6.4. 구현 환경

구현 환경은 소프트웨어적인 부분과 하드웨어 적인 부분으로 나눠서 설명하겠다.

6.4.1. 소프트웨어

운영체제 - Window XP SP2
HLA/RTI - MAK RTI (C++)
X3D SDK - Xj3D Toolkit (Java)
N/W Protocol (Viewer – RTI) - UDP

6.4.2. 하드웨어

VGA – Nvidia GeForce 6600
RAM – 1GB
CPU – Intel P4, 3.0GHz
입력장치 – Microsoft SideWinder Precision 2 Joystick
가상환경 공간 – 8ch iCAVE (6.2.3.4 참조, [Fig. 6-13] 참조)
모션 체어 - 2자유도 Joychair (6.2.4.1 참조, [Fig. 6-15] 참조)

6.5. 실험 결과의 평가

X3D를 이용하여 잠수함 시뮬레이션을 다채널 디스플레이로 하고, 모션 플랫폼을 연동하여 운동감을 추가하였다. [Fig. 6-16]는 그 구현 결과를 보여주는 그림이다. 이 시뮬레이션에서는 잠수함의 가속, 감속이 가능하고, 여러 가지 뷰포인트를 가지고 있어서 다양한 각도에서 잠수함의 시뮬레이션을 가시화 할 수 있다. 또한 잠수함이 움직임에 따른 선형 가속도, 원심력, 잠수함의 기울어짐, 어뢰 폭발후의 진동 등을 모션 플랫폼을 통해서 표현할 수 있다. 또한 본 연구에서는 시뮬레이션의 framework로 일반적인 TCP나 UDP를 사용하지 않고, HLA를 사용함으로써, 추가로 서로 다른 환경에서 개발된 시뮬레이션들을 확장적으로 붙여나갈 수 있다.

본 연구에서 X3D 파일 포맷을 사용한 Xj3D와 고가의 상업용 툴인 Paradigm社의 Vega, 그리고 중저가의 상업용 S/W인 Right Hemisphere社의 Deep Exploration CAD Edition의 몇 가지 항목에 대해서 비교해 보았다.

6.5.1. Frame Rate 비교

같은 3D 모델을 여러 뷰포인트에서 보았을 때의 Frame Rate를 각각 비교해 보았다. 그 결과를 [Table. 6-1]에 나타내었다.

Xj3D와 Deep Exploration의 경우 첫 번째 뷰를 제외하고는 거의 비슷한 성능을 보였다. 폴리곤의 수가 상대적으로 적은 View 1에서 Xj3D가 Deep Exploration 보다 Frame Rate이 높은 정확인 이유는 설명하기 어렵지만, Xj3D는 공개된 프로젝트로 개발이 되었기 때문에 기본적인 Shading 이나 렌더링에 초점이 맞추어 져있고, Deep Exploration은 상업용 툴이어서 화질이 높은 이미지를 보여주기 위해 여러 가지 효과들을 추가했기 때문에 Xj3D의 렌더링 속도가 더 빠를 수 있음을 짐작할 수 있다. Vega의 경우 전체적으로 볼 때 Frame Rate가 상대적으로 높음을 알 수 있다. 그 이유는 Vega는 내부적으로 스스로 LOD를 적용하여 멀리 있는 모델에 대해서는 화면에 표시를 하지 않음으로써 한 프레임당 보여주어야 하는 폴리곤의 개수를 줄일 수 있어서 Frame Rate가 높다는 것을 알 수 있다. 그러므로, Xj3D에 LOD를 잘 적용시키면 View 1에서와 같이 오히려 Vega와 같은 상업용 툴보다 더 빠른 결과를 얻을 수 있을 것이다.

6.5.2. Loading 시간 및 파일 크기 비교

[Table. 6-1]에서 볼 수 있듯이 Xj3D는 30초, Vega는 80초, Deep Exploration은 20초의 로딩시간을 갖는 것을 확인할 수 있다. 일반적으로 Loading 시간은 파일 포맷의 크기와 연관이 있다. [Table. 6-2]를 보면 같은 모델에 대한 서로 다른 파일 포맷의 크기 비교를 했는데, 이것을 참조하면, X3D가 VRML이나 Open Flight 보다 파일 크기가 더 작다는 것을 알 수 있다. 일반적으로 X3D는 VRML과는 다른 Encoding 방식을 사용하기 때문에 X3D가 VRML보다 더 최적화 된 파일 구조를 가지는 것을 알 수 있다. 하지만, X3D가 Open Flight 보다 더 최적화된 파일 구조를 가진다고 말하기는 조금 어렵다. 비록 binary 형태의 X3D 파일 크기는 Open Flight 보다 더 작지만, Open Flight는 내부적으로 X3D보다 더 많은 정보를 가지고 있을 가능성이 많고, X3D로의 변환 과정에서 기하 정보를 제외한 정보들이 손실되었을 가능성도 있기 때문이다. Deep Exploration은 같은 Open Flight 파일을 읽어 들이는데 걸리는 시간이 Vega보다 더 작은 걸로 봐서 내부적으로 최적화된 로딩 알고리즘을 사용하고 있음을 예측할 수 있다.

6.5.3. 개방성 비교

X3D는 2003년에 국제표준으로 채택이 되었고, 지금도 계속적으로 Open 된 프로젝트 형태로 개발이 진행되고 있기 때문에 사용하는 것에 비용이 전혀 들지 않는다. 또한, X3D를 가시화 하는 Viewer들도 현재는 추가 비용 없이 사용할 수 있다. Vega는 국내에서 2천만원 이상의 고가의 상업용 툴이고, Deep Exploration 또한 $1495 정도의 비용이 든다([Table. 6-1] 참고).

X3D는 모든 플랫폼에서 아무런 제약 없이 쓸 수 있다. 반면에 Vega는 Windows, Linux, SGI Irix 등의 플랫폼에 대해 각각 다른 버전들을 제공하고 있지만, 서로 호환이 전혀 되지 않고, 각각의 버전에 대해서 비용을 따로 지불해야 하는 문제점이 있다. 또한 Deep Exploration은 오로지 Windows에서만 사용이 가능하다.

6.5.4. 이미지 Quality 비교

역시 [Table. 6-1]를 보면 Image Quality에 대한 비교가 나온다. Xj3D와 Vega는 모두 깜박임(Flickering)이 일어나는 것을 볼 수 있다. 주로 일어나는 깜박임은 바다물과 육지가 겹쳐

져 있는 곳에서 일어난다. 또한 가까이서 볼 때 보다는 아주 멀리서 바라볼 때 나타난다. 그 이유를 살펴 보면, 물과 육지의 높낮이가 차이가 조금밖에 나지 않기 때문에 아주 멀리서 바라보게 될 경우 전체적인 스케일이 너무 커져서 상대적으로 육지와 바다물의 높이 차가 작아지게 된다. 이 경우 그래픽 하드웨어의 Z-buffer에 주어진 Depth의 차의 정밀도 보다 더 작아지면 그래픽 하드웨어는 두 개의 면 즉, 육지와 바다가 모두 같은 높이에 있는 걸로 착각을 하게 되어서 Hidden Surface Removal 또는 Occlusion Culling 을 못하게 된다. 그러므로 육지와 바다가 번갈아 가면서 보이기 때문에 깜박임 현상이 일어나는 것이다. 반면 Deep Exploration의 경우 Xj3D나 Vega에 비해 깜박임이 상대적으로 적게 나타나는 것을 볼 수 있다. 이것은 Deep Exploration이 Xj3D나 Vega보다 더 높은 정밀도를 사용하기 때문이다. 이러한 문제를 해결하기 위해서는 정밀도를 높이는 것이 가장 근본적이고 확실한 해결책이지만, Xj3D의 소스를 고치거나 Vega의 벤더가 정밀도를 높이지 않는 이상은 해결하기 어려운 부분이다. 하나의 편법으로는 육지와 바다의 높낮이 차이가 많이 나도록 육지를 높이든지 해수면을 낮추는 방법이 있다. 실제로 해수면의 높이를 낮추어서 실험을 한 결과 깜박거림이 사라지는 것을 알 수 있었다.

또한 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. X3D를 이용할 경우 이와 같은 효과를 나타내기 위해 Shader에 관한 연구가 필요하다. 하지만, X3D에서도 바다물의 투명도를 조절하고, 안개효과를 주어서 흐리게 만들고, 텍스쳐를 입힘으로써 어느 정도 비슷한 효과를 충분히 낼 수 있다.

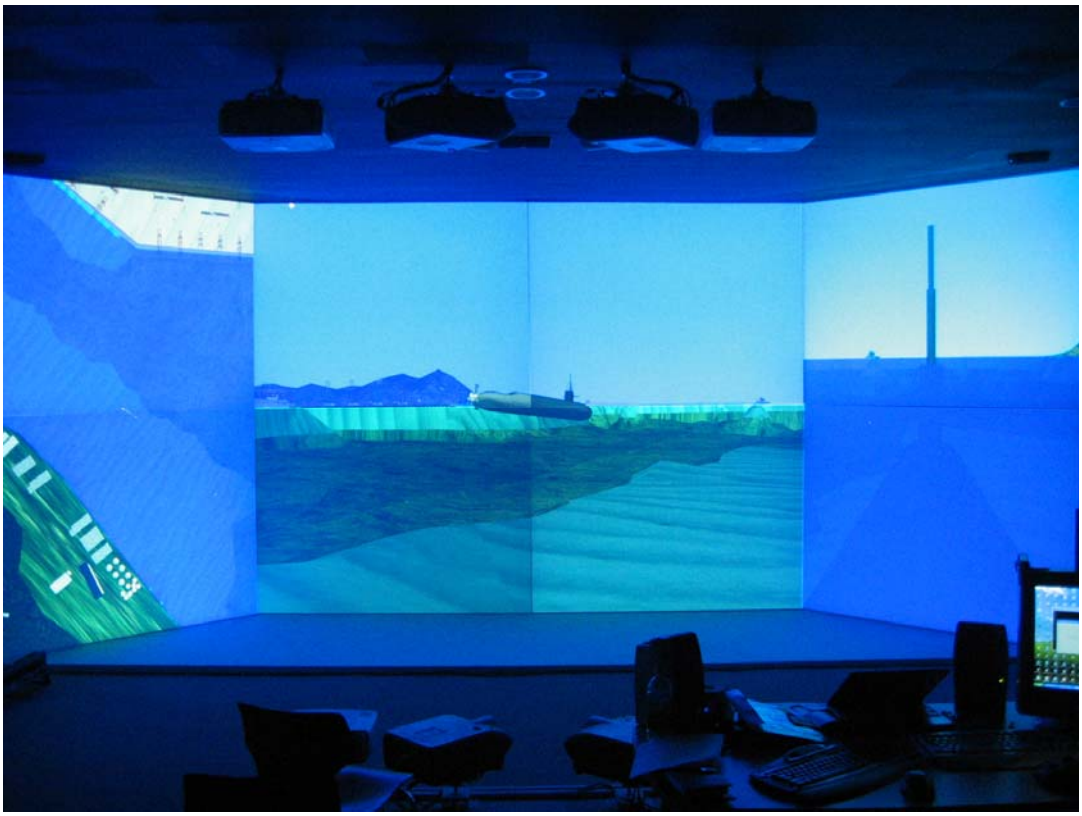

Fig. 6-16 X3D를 이용한 잠수함 시뮬레이션의 다채널 가시화 및 모션 플랫폼의 HLA/RTI 기반 통합 구현 화면

		Xj3D (X3D)	Vega	Deep Exploration (FLT)
FPS	View 1	30 fps	18 fps	11 fps
	View 2	10 fps	12 fps	10 fps
	View 3	8 fps	24 fps	8 fps
	View 4	10 fps	24 fps	9 fps
Loading Time		30 s	80 s	20 s
Open	Cost	0 $	More than 20000 $	1495 $
	Platform	All	Win, Linux, SGI Irix	Windows
	Etc	Compile only once	Pay for each Compile for each	High Quality for Static Display
Image Quality		**깜박거림이 심함** (Precision **낮음**)	**깜박거림이 심함** (Precision **낮음**)	**깜박거림이 덜함**(Precision **높음**) **화면자체를 그리지 않음**
		면 사이의 간격을 임의로 띄움으로써 해결		

Table. 6-1 실험결과 비교

파일이름	X3D		WRL	FLT
	ascii	binary	ascii	binary
Pusan_N4	14,887	1,347	27,395	12,598
Pusan_underwater	939	144	2,602	1,222
ohio_sm	386	64	1,116	407
Mk48Torpedo	25	6	73	N/A
Pusan_water	22	5	35	83
Pusan_New	1	0.5	1	11

Table. 6-2 포맷 별 파일크기 비교

7. 결론

7.1. 연구 요약

잠수함은 은밀성과 위협성 등을 가지고 있기 때문에 좋은 무기체계이다. 그렇기 때문에 군사들에게 있어서 잠수함 훈련은 아주 중요하지만, 실물 잠수함이 고가이고, 지역적, 시간적 제약을 가지고 있어서 M&S의 적용이 필요하다. 하지만, 기존의 잠수함 M&S는 고가의 상업용 가시화 툴과 전용 하드웨어를 사용함으로써 시스템을 관리하기 위한 특별한 장소의 문제, 유지 보수의 문제, 정해진 시간과 장소에서의 사용문제 등의 이유로 활용도가 떨어진다. 또한 단일채널 가시화로 인하여 몰입감이 떨어진다. 그리고 서로 다른 환경에서 만들어진 시뮬레이션들을 통합하기 어려운 문제도 많이 발생하였다.

이러한 문제점들을 해결하기 위해서 본 논문에서는 플랫폼 독립적이고 개방된 국제 표준인 X3D를 이용하여, 일반 사양의 PC를 사용하여 잠수함 시뮬레이션을 가시화 하였고, 몰입감 증대를 위해서 다채널 가시화와 모션 플랫폼을 적용하였다. 또한, 이들 모션 플랫폼, 조이스틱, 다채널 가시화 등 다른 환경에서 개발된 시뮬레이션 모듈들을 쉽게 통합하기 위해서 HLA/RTI를 적용하였다.

또한 이들을 실험적으로 구현하고 부산항 환경에서의 실험을 통해, 국제표준인 X3D가 아직 브라우저들이 개발 단계에 있지만, 상업용 시스템들과 비교할 만한 성능을 보이고 있으며, 일반 PC를 이용해 구축한 저가형의 시스템을 잠수함 M&S에 적용 가능함을 알 수 있었다.

7.2. 연구의 기여도

잠수함 M&S에 있어서 기존의 고가의 상업용 가시화 툴과 장비를 활용한 시스템과 본 연구에서의 구현한 플랫폼 독립적이고 개방된 국제표준을 사용하고, 저가의 일반 사양 PC를 활용한 시스템을 비교해 볼 때, 성능 상에 있어서 큰 차이를 보이지 않음을 알 수 있었다. 이럴 경우 저가의 시스템이라는 장점을 활용하여 시설을 많이 확충하여 해군 장병들이 잠수함 M&S의 환경에 더 쉽게 접근할 수 있고, 유지 보수가 쉬워 새로운 기술 발전에 대응하기 쉬우며, 해군의 전투력 향상에 도움을 줄 것이다.

또한, 단일채널의 한계를 넘을 수 있었다. 즉, 기존의 모니터와 같은 단일 채널을 활용할 경우 사용자가 직접 작업 환경 안에 있는 듯한 몰입감을 주기 못하기 때문에 제대로 된 운용 기술 습득에 어려움을 줄 수 있다. 그러나, 본 연구에서 다채널 가시화와 모션 플랫폼을 적용함으로써 실제 환경과 같이 몰입감을 증대시킬 수 있다.

그리고, 서로 다른 환경에서 개발된 시뮬레이터들은 일반적인 TCP나 UDP를 사용해서는 통합하기 어려운 점들이 많았다. 하지만, HLA/RTI를 적용해 봄으로써 상호호환성 뿐만 아니라 재사용성을 높일 수 있다.

또한, 기계공학의 응용 분야인 로봇이나 가상생산공장 시뮬레이션 등에 응용 할 경우 의미정보를 가지지 못하는 VRML의 단점을 극복하여 좀 더 완성도 높은 시뮬레이션이 가능하므로, 설계의 현장 적용에 필요한 시간과 비용 등이 줄어들 것으로 예상된다.

마지막으로, 본 연구에서 통합한 시스템은 앞으로 XMSF(eXtensible Modeling and Simulation Framework)의 Web-Enabled RTI를 적용할 경우 웹 상에서도 적용이 가능할 것으로 기대된다[54].

7.3. X3D 표준에 대한 개선 요구사항

본 연구에서 X3D를 사용하여 시뮬레이션의 가시화를 구현하여, X3D가 M&S를 위한 좋은 그래픽 파일 포맷임을 확인할 수 있었다. 그리고 X3D의 장점들 중에서 시뮬레이션에 사용된 기능의 하나는 충돌체크 부분이었다. X3D에 충돌 체크를 하는 노드인 “Collision”을 추가하게 되면 X3D Viewer가 자체적으로 충돌체크를 해 주었다. 하지만, 충돌체크를 할 때 추가적인 정보를 알 수가 없다는 단점이 있었다. 즉, 물체가 충돌을 했을 때 물체가 어느 방향으로 힘을 받는지에 대한 정보를 알 수가 없다. 그러므로, 물체가 충돌했을 때 힘의 Normal 벡터를 제공하는 필드가 추가되어야 할 것이다.

7.4. 향후 연구

본 연구를 통해서 X3D가 잠수함 M&S의 가시화에 활용될 수 있음을 보았다. 하지만, 여전히 고가의 상업용 툴과 비교를 했을 때, 이미지 Quality 등의 부분에서 미약한 부분이 나타남을 알 수 있었다. 그러므로 고급기법을 사용하기 위해서 X3D에서 Shader 기법을 활용하

는 추가 연구가 필요할 것이다. 그럴 경우 바다 물의 파도와 같은 표현이나 빛의 굴절 등을 잠수함 M&S 가시화에 쉽게 적용할 수 있다.

또한, 앞에서 Vega의 Frame Rate이 상당히 높은 이유가 LOD를 내부적으로 잘 적용했기 때문이다. 그러므로 X3D의 모델링에 최적화된 LOD를 적용함으로써 높은 Frame Rate를 얻을 수 있을 것이다.

본 연구에서는 한 대의 잠수함을 적용한 단순한 분산 시뮬레이션 환경을 구축하였다. 그 결과 구현에 있어서의 어려움은 없었지만, 현실감과는 약간 거리가 있는 구현이었다. 이에, 향후에는 여러 대의 잠수함과 여러 대의 비행기 및 전함 등을 배치하고 실재로 교전을 하는 등의 좀 더 대규모의 시스템을 구축함으로써, HLA의 수행 능력이나 현실적인 문제점들을 파악할 수 있을 것이다.

본 연구에서는 가시화 부분에 있어서는 공개된 국제표준을 사용하여 저가의 시스템을 구축할 수 있었지만, 분산환경시스템을 구성하는 Framework로서 고가의 상업용 MAK RTI를 사용하였다. 지금 현재 XMSF[54]에서는 Web 기술을 적용하여 Web-Enabled RTI를 제안하고 있으며, 계속 활발한 개발이 진행중이다. 그러므로 향후 Web-Enabled RTI의 기술이 완성도가 높아질 경우 MAK RTI 대신 Web-Enabled RTI를 사용함으로써, 분산환경시스템을 구성하는 Framework도 저가의 시스템으로 구성이 가능할 것이다.

참고문헌

1. 김창식, "Development of Virtual Ocean Environment System for Interactive Operational Use", Coastal and Harbor Engineering Research, Annual Report 2003, pp48~51, 2003
2. 원광연, 박재희, "감성공학과 가상현실", 한국정밀공학회지, Vol18, No. 2, pp40~45, 2001
3. Mike Hurwicz, "Web Virtual Reality and 3D in VRML or XML?", "http://www.webdevelopersjournal.com/articles/virtual_reality.html", 2000
4. Don Brutzman, "The Virtual Reality Modeling Language and Java", ACM, vol. 41, No. 6, pp. 57~64, June 1998
5. Daniel Selman, "Java 3D Programming", MANNING, pp24~33, March 2002
6. Brian K. Christianson, "Comparison of Vega and Java3D in A Virtual Environment Enclosure", Master's thesis, Naval Postgraduate School, March 2000
7. 타카시 이마기레, "DirectX9 셰이더 프로그래밍", 한빛미디어, pp26~32, July 2004
8. Wolfgang F. Engel, "Beginning Direct3D Game Programming 2nd Edition", Prima Publishing, pp38~55, Jan 2005
9. Ron Fosner, "OpenGL Programming for Windows 95 and Windows NT", ADDISON-WESLEY, pp42, 1998
10. Josef Wolte, "Information Pyramids", Master's thesis, Graz University of Technology, Oct 1998
11. 김용식, "가상 공장 시뮬레이션을 위한 PC 클러스터 기반의 다채널 가시화 모듈의 설계와 구현", 제 13회 HCI, CG, VR, Design, UI 학술대회 논문발표집 1-1, pp544~548, 2004
12. sgi社 싸이트, "http://www.sgi.com"
13. Open Scene Graph 싸이트, "http://www.openscenegraph.org"
14. Rynson Lau, Daniel Thalmann, "Emerging Web Graphics Standards and Technologies", Web Graphics Tutorial, IEEE, Feb 2003
15. 문홍일, "SEDRIS를 이용한 디지털 생산 시뮬레이션 환경의 융합", 한국시뮬레이션학회논문지, 14(2), pp15~24, 2005

16. Curtis Blais, Don Brutzman, “Web-Based 3D Technology For Scenario Authoring And Visualization : The SAVAGE Project”, Proceeding of I/ITSEC, 2001
17. Web3D Consortium, “http://www.web3d.org/x3d”
18. Octaga 싸이트, “http://www.octaga.com”
19. Curtis Blais, Don Brutzman, “Emerging Web-Based 3D Graphics For Education And Experimentation”, 2003
20. Edward M. Sims, William Y. Pike, “Reusable, Lifelike Virtual Humans For Mentoring And Role-Playing”, I/ITSEC, No 1611, p1~11, 2004
21. 이동훈, “X3D 가상환경에서의 확장 가능한 상호작용”, 한국정보과학회, 봄 학술발표논문집, Vol.30, No.1, pp349~351, 2003
22. 이성태, “X3D를 이용한 Humanoid 모델링과 애니메이션 기법에 관한 연구”, 한국해양정보통신학회, 추계종합학술대회지, 제6권, 제2호, pp812~816, 2002
23. H-Anim 싸이트, “http://www.h-anim.org”
24. Ivan Chang Kok Ping, “High Level Architecture Performance Measurement”, Master’s thesis, Naval Postgraduate School, 2000
25. 이성준, “수중운동체의 분산 시뮬레이션을 위한 HLA 기반의 모델 구조 연구”, 석사학위논문, 서울대학교, 2006
26. Singhal, Sandeep, “Networked Virtual Environments”, ACM Press, New York, NY, 1999
27. Dahmann Judith, Kuhl Frederick, “Creating Computer Simulation Systems – An Introduction To The High Level Architecture”, Prentice Hall, 1999
28. Ying Li, Ken Brodlie, “Web-Based VR training Simulator For Percutaneous Rhizotomy”, 2000
29. Jonathan C. Robert, Rob Knight, “Multiple Window Visualization On the Web Using VRML and the EAI”, Proceeding of 7th UK VR-SIG Conference, pp149~157, 2000
30. Chad F. Salisbury, Steven D. Farr, “Web-Based Simulation visualization Using Java3D”, Winter Simulation Conference, pp1425~1429, 1999
31. 이재철, “웹 상에서 STEP 조립체 모델의 가시화”, 한국CAD/CAM학회지, 1999학술발표회논문집, Network-Based System, pp295~299, 1999
32. Jack Ogren, “EAGLE and the High Level Architecture”, DMSO

33. DMSO의 HLA 싸이트, “https://www.dmso.mil/public/transition/hla/”
34. DMSO 싸이트, “http://www.dmso.mil”
35. Blaxxun 싸이트, “http://www.blaxxun.com”
36. Cosmoplayer 싸이트, “http://www.karmanaut.com/cosmo/player/”
37. Flux 싸이트, “http://mediamachines.com”
38. Xj3D 싸이트, “http://www.xj3d.org”
39. NIST VRML to X3D변환 툴 싸이트, “http://ovrt.nist.gov/v2_x3d.html”
40. Xj3D 개발 상태 보고 싸이트, “http://www.xj3d.org/status.html”
41. Mak Technologies, “Mak VR-Link Developer’s Guide”, Ver 3.7.2, 2002
42. 이창민, “ 동역학해석 기반의 선박운항 M&S 구현을 위한 HLA개념의 복합연동 Simulation설계”, 박사학위논문, 서울대학교, 2005
43. Right Hemisphere社, “http://righthemisphere.com”
44. Don Brutzman, Mike Zyda, Mark Pullen, Katherine L. Morse, “Extensible Modeling and Simulation Framework, JFCOM Exercise Opportunities”, JFCOM Workshop, 2003
45. Leandro Motta Barros, “A Short Introduction to the Basic Principles of the Open Scene Graph”, “http://www.cscience.org”, 2005
46. CAVE 싸이트, “http://www.evl.uic.edu”
47. Euler Angle 싸이트, “http://www.euclideanspace.com/maths/geometry/rotations/euler/index.htm”
48. Rotational Matrices 싸이트, “http://www.euclideanspace.com/maths/algebra/matrix/index.htm”
49. Quaternions 싸이트, “ http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/index.htm”
50. Axis-Angle 싸이트, “ http://www.euclideanspace.com/maths/geometry/rotations/axisAngle/index.htm”
51. Euler to Axis-Angle Conversion 싸이트, “http://www.euclideanspace.com/maths/geometry/rotations/conversions/eulerToAngle/index.htm”
52. SEDRIS 싸이트, “http://www.sedris.org”

53. Java3D 싸이트, “http://java3d.j3d.org/”
54. XMSF 싸이트, “https://www.movesinstitute.org/xmsf/xmsf.html”

감 사 의 글

부족한 저를 iCAD연구실의 한 일원으로 받아 주시고, 학문적으로, 인간적으로 이끌어 주신 한순흥 교수님께 감사의 말씀을 전합니다. 한순흥 교수님을 만난 것은 저의 인생에 있어서 커다란 보물의 발견과도 같은 것이었습니다. 부족한 제가 석사 학위를 받을 수 있도록 이끌어 주셔서 정말 감사드립니다. 그리고 바쁘신 와중에서도 부족한 저의 논문을 심사해 주신 권동수 교수님, 김정 교수님, 그리고 이창민 박사님 정말 감사드립니다.

인생은 혼자서 사는 것이 아니라는 것을 느끼게 된 것은 iCAD 연구실의 멤버들 때문이었습니다. 제가 할 수 없었던 많은 부분들을 연구실 선후배님들의 도움으로 해낼 수 있었고, 이렇게 졸업을 할 수 있었습니다. 함께 연구실에서 밤을 새면서 많은 조언과 도움을 주셨던 병현이형, 항상 따뜻하고 힘이 되는 말씀을 해 주셨던 상욱이형, 같은 방에서 생활하며 논문 구현에 큰 도움을 주신 종환이형, 삶에 지혜를 주시던 진상이형, 아낌없는 충고와 조언을 해 주셨던 병철이형, 친형같이 편하게 대해 주셨던 효광이형, 연구실 동기이자 최종발표에 많은 도움을 주었던 일환이, 후배이면서도 선배인 나보다 아는 것이 많으면서도 겸손한 수철이, 매력적인 중국인 아가씨 Danhe, URP 과정으로 연구실에 있는 승훈이, 일처리를 부탁할 때마다 깔끔하게 처리해 주셨던 진아씨, 그리고 부품디비의 양희경씨, 안미라씨, 노선진씨, 귀목이, 김화수씨, 장광섭 과장님, 김준호 이사님께, 그리고 지금 이름을 적지 못한 많은 분들께 감사의 말씀을 드립니다.

2004년 9월에 함께 입학한 몇 안 되는 입학 동기인 경록이형, 민규형, 원익이에게도 감사의 말씀을 전합니다. 지칠 때마다 안식처가 되었던 한양대 기계공학부 동기들과 동문들에게 감사의 말씀을 전합니다. 28년 동안 길러주시고 늘 아낌없는 칭찬과 눈물과 기도로 후원해 주셨던 부모님께 정말 많은 감사의 말씀을 드립니다. 특히 군대에서부터 지금까지 항상 내 곁에서 동고동락 하였던 저의 아내 효진이에게 감사를 드립니다. 마지막으로, 내 삶의 모든 부분을 인도해 주셨던 하나님께 감사 드립니다.

이 력 서

성　　명 : 허 필 원 (許 弼 援)

생년월일 : 1979년 9월 12일

출 생 지 : 부산광역시 서구 남부민동

본　　적 : 경상북도 경산시 남산면 전지리

학　　력

1995.3 ~ 1998.2 : 부산고등학교

1998.3 ~ 2004.8 : 한양대학교 기계공학부 (B.S.)

2004.9 ~ 2006.8 : 한국과학기술원 기계공학과 (M.S.)

경　　력

2000.11 ~ 2003.1 : 대한민국 육군

학 회 활 동

한국 CAD/CAM 학회

한국 시뮬레이션 학회