석 사 학 위 논 문

Max/MSP와 OpenGL을 이용한 인터랙티브 음악 시스템 개발 연구

(멀티미디어음악작품 『NATUM.scene.2』를 중심으로)

지도교수 김 준

동국대학교 영상대학원
멀티미디어학과 컴퓨터음악전공
최 홍 찬

2 0 0 5

석 사 학 위 논 문

Max/MSP와 OpenGL을 이용한 인터랙티브 음악 시스템 개발 연구

(멀티미디어음악작품 『NATUM.scene.2』를 중심으로)

최 홍 찬

지도교수 김 준

이 논문을 석사학위논문으로 제출함.

2005년 12월 21일

최홍찬의 음악석사학위(컴퓨터음악전공) 논문을 인준함.

2006년 1월 일

위원장: 조 형 제 (인)

위 원: 엄 기 현 (인)

위 원: 김 준 (인)

동국대학교 영상대학원

목 차

그 림 목 차

수 식 목 차

I. 서 론

1. 연구 목적

1) 연구의 배경 및 동기

작품 『NATUM』[1]은 사용자의 조작에 따라 영상과 음향이 변화하는 멀티미디어 시스템이다. 이 작품의 제작에 있어 가장 중요한 동기는 멀티미디어가 가지는 장점을 백분 활용한 예술 작품을 만드는 것이며, 사용자의 조작에 대한 시스템의 반응이 어떠한 형태일 때 가장 효과적인가도 중요한 연구 배경이다.

둘째, 필자가 추구해 오던 디지털 시스템이 가지는 미학도 중요한 연구 동기가 되었다. 아날로그 시스템에서 규칙과 반복을 찾아내 그것을 디지털 시스템으로 모방하는 것, 그리고 일단 모방을 시작한 디지털 시스템이 아날로그 시스템과는 독립적으로 스스로 성장해가는 것들에 대해 심미안을 갖고 진지하게 접근하려는 의도가 있었다.

마지막으로, 기존의 「인터랙티브」(interactive)[2] 음악작품들이 보여주었던 영상의 미숙함을 탈피하여 섬세하고 세련된 영상과 음향의 조화를 이루어내는 것이 이 작품의 또 다른 중요한 연구 동기가 되었다. 기승전결의 구조를 미리 성립하여 시청자에게 일방적으로 그 내용을 전달하기 보다는, 상징적인 음향과 영상을 보다 세련된 방법으로 포장하여 시청자가 다양한 생각과 반응을 보일 수 있도록 유도하는 것이

---

1) 나툼(NATUM)은 순우리말로서 '나타내다'라는 동사의 명사형이며, 주로 수학 분야에서 많이 사용된다.

2) '상호간'의 뜻을 지닌 인터(Inter-)와 '활동적'의 뜻을 지닌 액티브(active)의 합성어로, 상호활동적인, 곧 쌍방향이라는 의미를 지닌다.

중요한 목표가 되었다.

2) 연구의 목적

이 작품을 위한 연구의 목적은 다음과 같다. 첫째, 사용자로부터 효과적인 반응을 얻어내기 위한 3D 컴퓨터그래픽과 음향을 조합하는 가장 이상적인 방법을 찾아내는 것이다. 비록 이 시스템은 디지털을 기반으로 하여 제작되었지만, 사용자가 그것이 디지털로 이루어져 있다는 사실을 반드시 알 필요는 없으며, 오히려 아날로그 시스템에 가까운 모습을 보여줄 때 더 효과적이라는 사실을 염두에 두었다. 실제에 가까운 3D 컴퓨터그래픽 애니메이션에 사람들이 열광하듯이, 단순한 컴퓨터그래픽이라 할지라도 그것에 사실적인 움직임이 부여되면 사람들의 반응은 달라진다. 그러한 이유로 3D 모델[3]들에 물리적인 움직임을 적용하였으며, 그에 따른 수많은 수학적 알고리듬[4]과 물리적 분석이 필요했다.

둘째, 누구나 손쉽게 사용할 수 있는 직관적인 사용자 인터페이스를 제작하는 것이다. 필자가 생각하는 「인터랙티브」 시스템이라는 것은 특정인이 아닌, 시스템에 접근하는 모든 사람들에게 충분한 「인터랙티비티」(interactivity)[5]를 제공할 수 있는 시스템이며, 시스템을 속속들이 알고 있는 사람만이 소리를 낼 수 있고 영상에 변화를 줄 수 있는 시스템이라면 사실 큰 의미가 없다고 생각한다. 따라서 사용자 인터페이스는 단순하고 직관적이어야 하며 그것을 조작하는 재미도 있어야 한다.

---

3) 3차원 그래픽을 이루는 원시 도형(primitives)
4) 잘 정의되고 명백한 규칙들의 집합 또는 유한 번의 단계 내에서 문제를 풀기 위한 과정
5) 상호작용성.

마지막으로, 개방된 형태의 멀티미디어 음악작품을 제작하는 것이다. 순간적인 연출과 일회성뿐인 음악작품이 아니라, 언제든 그 예술적 구성이나 구조가 바뀔 수 있고, 필요에 따른 수정과 재구성이 손쉬운 작품이 되어야 한다. 이는 기승전결을 가진 선형적(linear) 구조를 가진 작품의 제작이라기보다는, 어떤 구조라도 유연하게 받아들일 수 있는 시스템을 제작하는 것을 의미한다.

2. 작품 배경

1) 작품의 예술적 배경

디지털은 '상징(representation)'이다. 비트[6]가 모여 바이트[7], 워드[8]가 되고, 이러한 기호들의 나열은 결국 실재하는 어떤 것의 의미를 상징하게 된다. 디지털 시스템은 '상징'이라는 방법론을 이용하여 아날로그 시스템을 끊임없이 모방하고 있으며, 그와 함께 인간이 누릴 수 있는 다양한 형태의 디지털 기술이 제안되고 발전되어 왔다.

하지만, 디지털은 아날로그의 부분집합만으로 치부되기에는 너무나 많은 미적 요소들을 갖고 있다. 디지털을 아날로그의 그 무엇이 아닌 디지털 그 자체로서 바라보는 관점에서 이 작품은 시작되었다.

우리는 현실을 서술하기 위해 상징이라는 방법을 사용하지만, 결국 상징이라는 것은 그 무엇이라도 될 수 있는 가능성을 내포하고 있음이다. 요컨대, 디지털은 현실의 상징으로서 형성되지만, 이미 형태를 이룬 상징이 반드시 현실의 그 무엇을 가리킬 필요는 없다는 것이다. 종속관계

---

6) binary digit의 약자. 디지털 데이터의 최소 단위로 0 또는 1의 값을 갖는다.
7) byte. 8개의 비트가 모여 이루어진 데이터의 단위
8) word. 4개 또는 그 이상의 바이트가 모여 이루어진 데이터 단위

가 사라지고 상징은 상징 그 자체로서 독립하게 되는 것이다.

작품 『NATUM』은 그 자체로 표현이자 상징이다. 좁은 의미로는 컴퓨터 내부적으로 데이터를 표현하기 위한 데이터형[9](representation)을 일컫는 것일 수도 있다. 중요한 것은 어떤 것을 상징하고 있는가에 대해서는 말하지 않는다는 점이다.

이처럼 이 작품에서는 구체적인 주제를 제안하기보다 단순하고 모호한 형태의 영상과 음악, 음향, 그리고 사용자 인터페이스(user interface)를 제공함으로써 수동적 시청자 또는 능동적 시청자로 하여금 더 많은 것을 상상하고 생각할 수 있는 환경(environment)을 제시한다.

2) 작품의 기술적 배경

작품의 중심에서 가장 큰 역할을 수행하는 것은 Max/MSP이다. Max/MSP는 Cycling '74[10]가 개발한 응용프로그램(application)으로, 산술처리, 데이터처리, 미디 데이터처리, 음향처리, 간단한 화상처리 등을 위한 다양한 객체(object)[11]를 제공하며 사용자의 요구에 따라 객체를 사용하여 프로그래밍 할 수 있는 환경을 제공한다.

모든 작업을 실시간(real-time)으로 할 수 있으며, 그래픽유저인터페이스(GUI) 방식을 채택하고 있다는 장점이 있다. 때문에 누구나 손쉽게 사용할 수 있는 동시에, 「인터랙티브」한 작품을 만들기에 무척 용이하다.

중요한 역할을 수행하는 또 다른 소프트웨어는 Jitter이다. Jitter역시

9) 컴퓨터 공학 분야에서는 data type과 representation은 거의 같은 의미로 사용된다.
10) http://www.cycling74.com/
11) 특정 작업을 수행하기 위한 함수들을 포함하고 있는 기능적 집합체

Cycling '74사의 소프트웨어 패키지로 Max/MSP에 추가되는 형태로 설치되며, 다양한 영상 합성, 3D 그래픽 생성 등을 위한 기능을 제공한다.

Max/MSP와 Jitter는 기본적으로 그래픽유저인터페이스 기반의 프로그래밍 환경을 제공하나, 이 작품의 특성상 대단히 많은 양의 메시지(message)[12]들이 객체로 전달되며 이는 곧 컴퓨터 시스템의 성능 저하로 연결되므로, 그래픽유저인터페이스를 사용하는 대신 「자바 스크립트」(Java script)[13]를 도입하여 많은 양의 메시지를 효율적으로 전달할 수 있도록 했다.

마지막으로, 사용자의 물리적인 입력을 컴퓨터가 처리할 수 있는 디지털 데이터로 변환하기 위해 「디지타이저」(digitizer)[14]를 사용하였으며, 「디지타이저」는 Infusion Systems[15]의 I-CubeX Wi-minidig을 「블루투스」(Bluetooth)[16] 인터페이스를 통해 컴퓨터에 연결하였다.

사용자의 물리적 입력을 일차적으로 받는 부분은 센서로 제작했으며, I-CubeX 설정 소프트웨어를 사용하여 센서에 대한 여러 가지 사항을 설정했다.

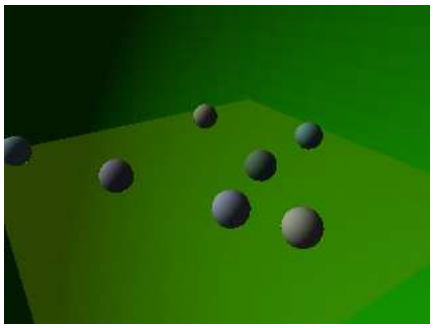
---

12) 객체에 내장된 함수를 호출하기 위해 전달되는 데이터
13) 미국의 Netscape Communication이 개발한 스크립트 언어
14) 아날로그 데이터를 디지털 형식으로 변환시키는 장치
15) http://www.infusionsystems.com/
16) 근거리에 놓여 있는 컴퓨터와 이동단말기·가전제품 등을 무선으로 연결하여 쌍방향으로 실시간 통신을 가능하게 해주는 규격을 말하거나 그 규격에 맞는 제품을 이르는 말

II. 본 론

1. 작품의 내용

작품 『NATUM』은 도입, 전개, 발전, 절정, 해결의 다섯 부분으로 구성되어 있으며, 이 다섯 개의 부분은 영상과 음향, 음악, 무대 조명의 양상이 서로 다르게 나타나도록 제작되었다.



[그림 1] 도입부분의 장면

도입에 해당하는 첫 번째 부분은 평면 위에 놓인 공(구, sphere)들이 자연스럽게 발생하여 움직이고, 서로 충돌하여 다양한 변화를 만들어 내도록 설계되었다. 도입 부분으로서의 충분한 역할을 위하여 느리고 섬세한 화면의 움직임에 초점을 맞추어 진행되도록 했다. 사용자의 조작에 의해 공들이 움직이며 서로 충돌할 때 마다 합성음과 테이프음악(tape music)[17]이 생성 및 재생된다.

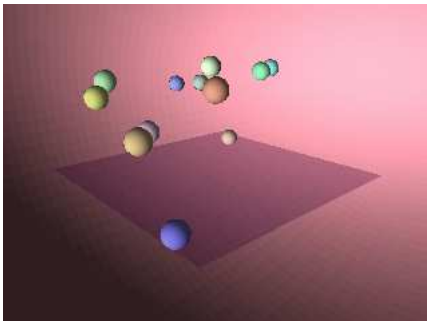
무대 조명의 컬러는 도입 부분의 컬러와 일치 하는 녹색 계열이며, 연주자를 직선으로 내리 비추는 핀(pin) 조명[18] 형태로 설정했다.

---

17) 미리 녹음하여 준비해 둔 음악

전개인 두 번째 부분은 평면의 수직 위 상공에서 공들이 자유 낙하하며 평면과 충돌하면 탄성에 의해 튀어 오르도록 설계되었다. 중요한 것은 도입 부에 비해 섬세하면서도 보다 역동적인 느낌을 이끌어내는 것이었다. 도입부분에서는 충돌과 관성을 핵심적인 움직임으로 사용했다면, 전개부분에서는 낙하와 탄성을 그것으로 사용했다. 첫 번째 부분과 마찬가지로 사용자의 조작에 의해 공들이 발생하여 낙하하며 충돌할 때 마다 합성음(synthesized sound)과 테이프음악이 생성 및 재생된다.

무대 조명은 화면의 전체적인 색상과 일치하는 분홍색에서 연한 자주색 사이의 색상이며, 도입부분과 마찬가지로 핀 조명을 사용했다.

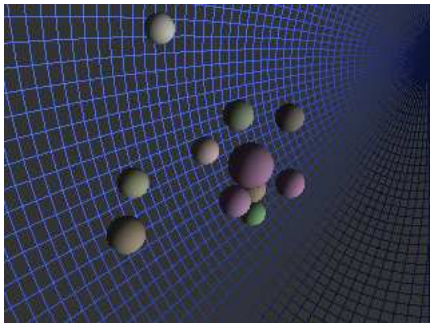


[그림 2] 전개부분의 장면

세 번째 부분은 중력이나 충돌이 없는 비현실적인 공간으로 공들은 자유롭게 상하좌우로 운동을 한다. 사용자가 체계를 자극할 때 마다 구들의 움직임은 더더욱 거칠어진다. 그리고 곧 다가올 절정 부분의 역동적인 화면을 위하여, 그 전주에 해당하는 움직임을 보여준다. 사용자의 자극에 대한 반응도는 도입, 전개 때 보다 훨씬 크며, 이렇게 구들이 자극을 받아 변화를 일으킬 때 마다 합성음과 테이프음악이 생성 및 재생된다.

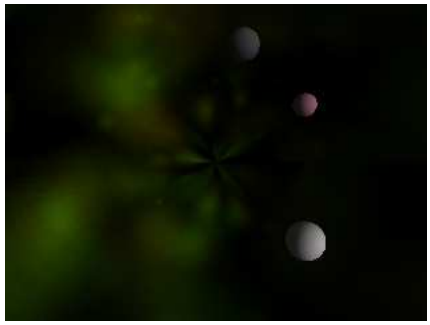
---

18) 피사체를 더욱 강조하기 위해 집중적으로 조명을 비추는 것



[그림 3] 발전부분의 장면

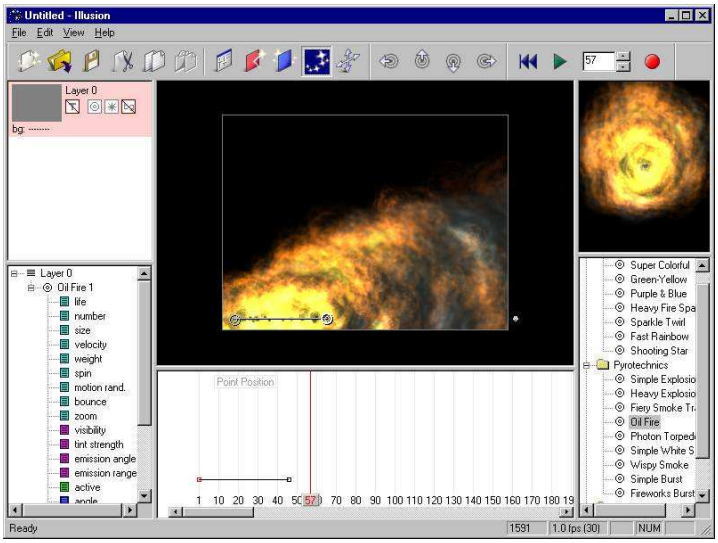
무대 조명은 어두운 파란색을 사용하며, 앞의 두 부분과 마찬가지로 핀 조명을 사용했다. 화면 자체의 느낌이 어두우므로 지나치게 조명이 밝아지지 않도록 주의했다.



[그림 4] 절정부분의 장면

네 번째의 절정부분은 이 작품의 최고조에 해당하는 부분으로, 추상적인 이물질들이 부유하고 있는 공간에서 공들이 쉴 새 없이 여러 방향으로 이동한다. 이 부분에서 사용자의 조작이 공들에 미치는 영향은 이전의 부분들만큼 즉각적이지 않다. 사용자의 입력에 대해 공들은 비행 방향을 천천히 변화시키는 반응을 보인다. 또한 자극에 대한 반응으로 저주파를 다수 포함하고 있는 합성음의 음량이 커지며, 음향의 변화로 얻게 되는 효과 또한 현재까지의 진행에서 가장 극적으로 드러난다.

절정 부분은 이 작품에서 유일하게 3D 모델에 텍스쳐 매핑(texture mapping)[19]이 사용되는 부분으로, Particle Illusion이라는 소프트웨어로 제작된 동영상이 3D 모델에 투영되었으며, 도입, 발전, 전개부와는 완전히 다른 느낌을 갖도록 만들어졌다.



[그림 5] 파티클 일루젼(particle illusion) 소프트웨어

무대 조명은 이제까지와는 다른 형태로 무대 전체를 추상적인 문양이 나타나는 조명으로 어둡게 비추었다. 다양한 색상이 나타나며 조금씩 그 형태를 변화해 가는 문양들이 영상의 모습과 일치되도록 했다.

마지막 다섯 번째 부분은 도입부와 동일하며, 원점으로 회귀된 체계의 모습을 표현하고 있다. 가장 정적으로 조작되어야 할 부분이며, 완결되지 않은 느낌을 가진 악곡에 맞추어 섬세하고 고요하게 작품이 완료되도록 한다.

지금까지의 각 부분에 대한 설명은 이 작품이 연주자에 의해 연주되

19) 컴퓨터 기법의 하나로, 표현하고자 하는 이미지나 물체의 사실감을 높이기 위해 그 표면에 원하는 무늬나 색을 입히는 작업

는 선형적 연주 작품일 경우에 대한 것이다. 선형적 연주 작품에 대한 시나리오는 본론 4의 3)에서 더 자세히 다루게 될 것이다.

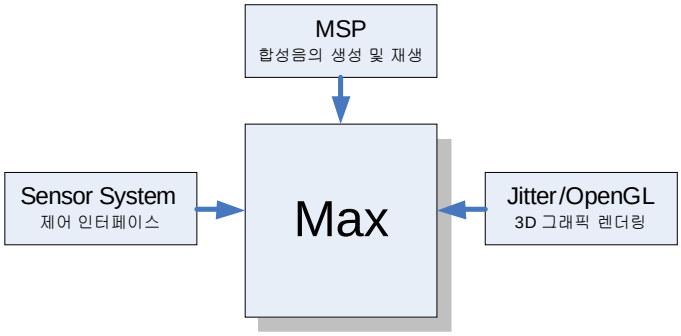
또한, 이 작품이 설치 형태로 일반 대중에게 제공될 경우에는 이와는 다른 양상으로 다양한 부분들이 설계되고 생성되어야 하는데, 무엇보다도 중요한 것은 일반 대중을 위한 작품의 경우에는 선형적인 시간 구성을 적용할 수가 없다는 점이다. 설치 형태의 시스템은 무작위에 가까운 사용자의 자극을 받아들일 준비가 되어있어야 하며, 자극의 형태를 더욱 적극적으로 분석하여 영상과 음향에 반영해야 한다. 설치형태의 작품에 대해서는 결론 3에서 보다 자세하게 이야기 할 것이다.

2. Max 환경을 통한 다양한 구성요소의 통합

서론의 2의 2)를 통해 이미 서술한 바 있지만, Max는 미디 데이터처리, 음향처리, 동영상처리가 가능하다는 커다란 장점이 있으며, 때문에 이 작품의 중추적 역할을 수행하고 있음을 다시 강조한다.

다양한 구성요소를 한 자리에 모아서 데이터의 흐름을 관리할 수 있고 각 구성요소 서로가 유기적으로 연결될 수 있기 때문에, 사용자의 입력에 대한 반응을 즉각적으로 내놓는 것이 무척 용이하다. 본 작품의 경우에는 사용자의 입력이 영상·음향으로 영향을 미치고 있지만, 필요에 따라 사용자 입력이 영상으로, 영상이 음향에게 영향을 미치도록 설계할 수도 있다.

Max가 강력한 도구인 또 다른 이유는 자바 스크립트 언어를 사용한 프로그래밍 기능을 자체적으로 지원한다는 것이다. 자바 스크립트는 본래 웹 사이트를 동적으로 조작하기 위해 만들어진 프로그래밍 언어이나, Max에 사용할 수 있는 자바 스크립트는 웹 사이트의 조작이 목



[그림 6] 작품의 기능적 구성도

적이 아니며 데이터 처리 및 수치 계산을 위해 사용할 수 있도록 약간 수정된 형태를 취하고 있다. 본 작품에서는 그러한 자바 스크립트의 기능을 백분 활용하여 컴퓨터의 데이터 처리 과정을 최대한 단순화 시켰다.

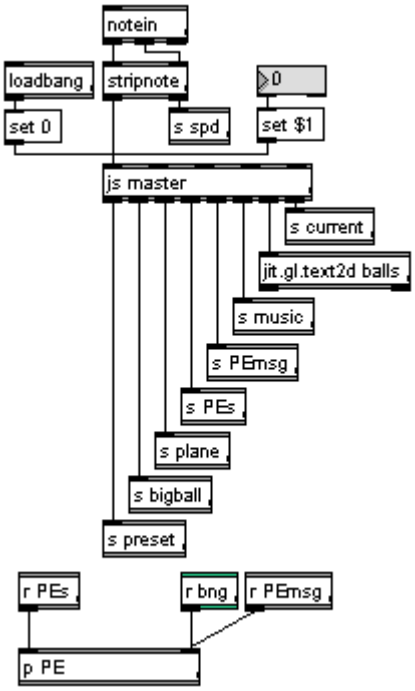
1) 중심제어부의 Max 프로그래밍

중심제어부는 MSP로 이루어진 음합성부분과 Jitter/OpenGL로 이루어진 3D 영상생성부분을 종합적으로 제어하고 있다. 물론 가장 중요한 역할 중 하나인 사용자의 입력을 받아 그것을 적절한 메시지로 변환하여 각 모듈로 보내는 일을 하도록 설계되어 있다.

자바스크립트로 이루어진 「master」 객체는 구체적으로 다음과 같은 일을 한다.

- 미디 데이터를 입력 받아 처리한 후, 객체들에게 메시지를 전달
- 도입, 전개, 발전, 절정, 결말의 각 단계에 해당하는 음악의 재생
- 3D 모델을 렌더링(rendering)[20] 하여 화면에 나타낼 것인가 결정

20) 물체의 모양을 그 형이나 위치, 광원 등의 외부 정보를 고려하면서 실감 나는 화상으로 표현하는 컴퓨터 도형 기법. 컴퓨터 내부에 기록되어 있는 모델 데이터를



[그림 7] 중심제어부의 Max 패치

- 3D 모델에 텍스쳐를 사용할 것인가 아닌가의 여부를 결정
- 2D 텍스트의 생성 및 소멸

다음은 master.js 스크립트 코드의 일부분이다.

```
// [프로그램 1] master.js
switch (p) {
     case 0:
        outlet(1, "drawto", "temp");
        outlet(1, "poly_mode", 0, 0);
        // 전체를 감싸고 있는 큰 공의 폴리곤 모드 설정
        outlet(1, "smooth_shading", 0);
        outlet(1, "lighting_enable", 1);
```

디스플레이 장치에 묘화(描畵)될 수 있도록 영상화하는 것을 말한다.

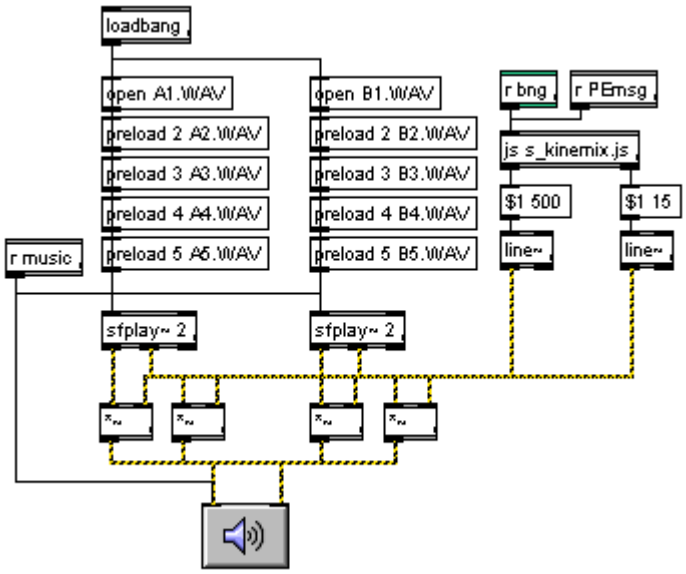
```
        outlet(1, "blend_enable", 1);
        outlet(1, "shape", "sphere");
        outlet(1, "scale", 60.0, 60.0, 60.0);
        outlet(1, "dim", 100, 100);
        outlet(2, "drawto", "temp");
        outlet(2, "smooth_shading", 0);
        outlet(2, "lighting_enable", 1);
        outlet(2, "blend_enable", 1);
        outlet(2, "shape", "plane");
        outlet(2, "rotate", 270, 1, 0);
        outlet(2, "scale", 12.0, 12.0, 12.0);
        outlet(2, "position", 0.0, -1.0, 0.0);
        outlet(4, "drawto", "temp");
        // 작은 공들을 화면에서 사라지게 함
        outlet(5, 0);
        // 음악 재생하지 않음
    break;
// end of script
```

2) MSP를 이용한 신디사이저(synthesizer)의 구현

작품 『NATUM』에서는 음을 직접 생성하거나 생성된 음에 또 다른 합성처리를 하는 방법을 최대한 사용하지 않으려 했다. 3D 컴퓨터그래픽이 작품의 핵심이기 때문에, 실시간으로 그려지는 3D 컴퓨터그래픽을 효율적으로 보여주기 위해 많은 처리능력을 필요로 하는 음합성은 피해야 했다.

따라서 소리의 작곡 및 합성은 외부 소프트웨어에서 테이프음악 형태로 만들어 둔 뒤, 다양한 형태의 테이프음악을 엠에스피의 「sfplay~」 객체를 사용하여 선택적으로 재생하도록 설계 되었다. 이 테이프음악들을 만들기 위해 사용된 여러 가지 기술적 사항들은 3장에 자세히 소개되어있다.

테이프음악으로 사용된 것은 크게 A와 B 두 가지로 나눌 수 있는데, A에는 자연스러운 피아노 음색이 녹음되어 있으며, B에는 피아노 음색을 소재로 한 합성음이 담겨져 있다.



[그림 8] 음합성을 위한 Max 패치

오토 밸런싱 믹싱 시스템(auto-balancing mixing system)이라 명명한 다음의 간단한 코드는 간단한 계산을 통해 테이프음악 A와 B의 음량이 부드럽게 오버랩(overlap)[21]되도록 만든다.

```
// [프로그램 2] s_kinemix.js
function bang() {
   if(vb > 0.0001) {
      // 테이프음악 B의 음량이 0.0001보다 클 때 다음의 계산을 수행
      vb *= gg;
      // gg는 중력가속도를 의미하며 테이프음악 B의 음량을 감쇄시킴
```

21) 한 장면에 다음 장면이 겹치면서 먼저 장면이 점점 사라지고 나중 장면이 점차 명확해지는 가운데 장면이 전환되는 기법. 여기서는 음향에 적용된 것을 의미한다.

```
      va = 1.0 - vb;
      // 테이프음악 A의 음량은 B의 음량에 반비례 하여 증가 또는 감쇄
      if(vb > 0.7) vb = 0.7;
      if(va > 0.7) va = 0.7;
      // 테이프음악 A, B의 음량은 0.7을 넘어설 수 없음
      outlet(0, va);
      outlet(1, vb);
   } else {
      vb = 0.0;
      // 테이프음악 B의 음량이 0.0001보다 작은 경우 0으로 내림
   }
}
// end of script
```

3) Jitter/OpenGL을 사용한 실시간 3D 렌더링

자바스크립트를 사용하여 자연스러운 값의 변화를 만들어 내는 것은 결국 자연스러운 애니메이션으로 연결된다. 게다가 이 자연스러움이 현실세계의 물리적인 요소를 지니고 있다면 보는 사람에게 더욱 인상적인 느낌을 줄 수 있다.

이 작품에서 움직이도록 만들어진 3D 모델들에 적용된 이동 방법은 다음의 수학적 공식에 기반을 두고 있다.

$$X_{next} = X_{cur} + S \times (X_{target} - X_{cur})$$

X_{next}; 다음 값　　X_{cur}; 현재 값

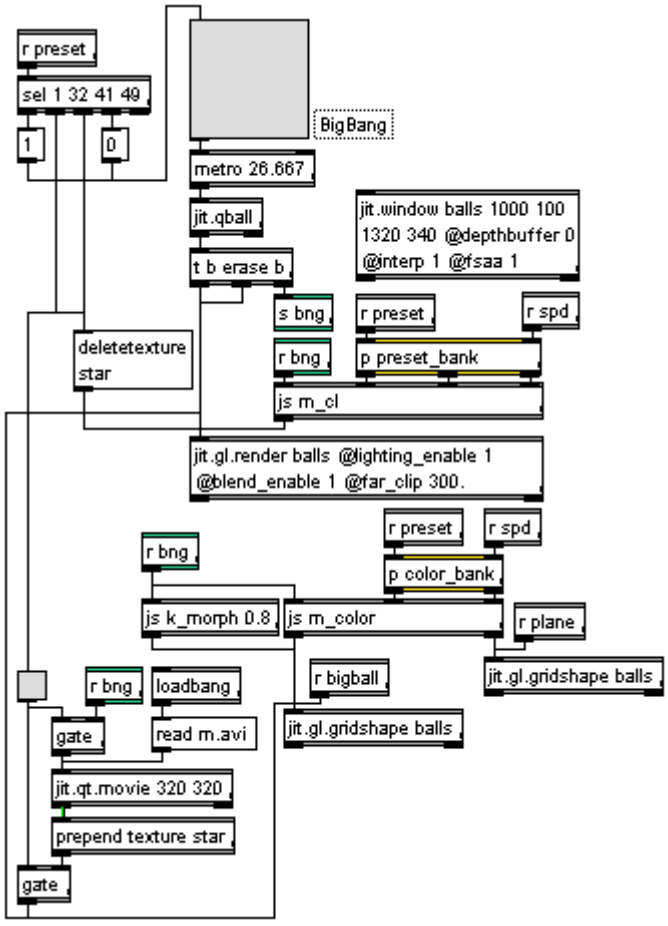
S; 변화 속도　　X_{target}; 목표 값

[수식 1] 부드러운 이동을 위한 수학공식

이 공식의 요점은 현재의 값 X_{cur}가 목표가 되는 값 X_{target}으로 변해갈 때, X_{cur}와 X_{target}의 중간에 있는 값들이 계산을 통해 얻어진다는 것이

다. 물론 이 공식만으로 그러한 중간 값들이 얻어지는 것은 아니다. 이 계산 과정이 X_{cur}가 X_{target}에 도달할 때 까지 반복적으로 이루어짐으로써 중간 값의 도출이 가능해지는 것이다.

이 연속적으로 얻어지는 중간 값들을 3D 객체의 좌표에 대입하면 자연스러운 물리적인 움직임을 가진 애니메이션을 볼 수 있게 되는 것이다.



[그림 9] 조명과 색상, 시점의 변화를 위한 Max 패치

```
// [프로그램 3] 부드러운 변화를 위한 코드 - js m_color, js m_cl
function itp1() {
   if(Math.abs(this.ta - this.a) <= 0.01) this.a = this.ta;
   // 계산량 감소 및 오버플로우 방지를 감소시키키 위한 비교문
   else this.a += this.spd * (this.ta - this.a);
   // 보간 공식
}
// end of script
```

이러한 중간값 산출 프로그램은 작품 전반에 걸쳐 사용되었으며, 특히 조명과 시점의 이동 및 3D 객체들의 색변화에 적용되어 부드럽게 변화 할 수 있도록 만들었다.

① 운동량 보존 및 충돌 검출 프로그램

3D 모델의 충돌 검출 및 이동은 도입부분의 연출에 커다란 역할을 하는 부분이다. 3D 모델의 좌표와 반지름을 기반으로 하여 두 3D 모델의 좌표 간의 거리가 반지름의 2배가 되는 것 보다 작은 경우, 두 모델이 충돌한 것으로 간주하고 그 후 3D 모델의 벡터(vector)[22]의 방향을 변화시킴으로서 충돌하여 튕겨 나가는 동작이 이루어진다.

```
// [프로그램 4] k_coll.js
var diffX = coll[j].x - coll[i].x;
var diffY = coll[j].y - coll[i].y;
var distance = Math.sqrt(diffX*diffX+diffY*diffY);
// 두 공의 중심 사이의 거리
if (distance == 0) distance = 1;
// 단위 벡터의 계산
```

22) 크기뿐만 아니라 방향을 가지는 양.

```
var unitDiffX = diffX / distance;
var unitDiffY = diffY / distance;
   // 반지름의 합이 중심사이의 거리보다 작은지의 여부를 판단(충돌 검출)
if(distance <= 2 * rad_c) {
   // B - Before, v - velocity, p - parallel, A와 B는 충돌한 두 공을 의미
   var BvpA = unitDiffX * coll[i].vx + unitDiffY * coll[i].vy;
   var BvnA = -unitDiffY * coll[i].vx + unitDiffX * coll[i].vy;
   var BvpB = unitDiffX * coll[j].vx + unitDiffY * coll[j].vy;
   var BvnB = -unitDiffY * coll[j].vx + unitDiffX * coll[j].vy;
   // 현재 충돌했더라도 다음번에 충돌할지 여부를 판단함
   if(BvpA - BvpB > 0) {
      // 운동량 보존 법칙에 따라 충돌후 두 공의 수평성분 속도 계산
      // A - after
      var AvpA = BvpA + (1 + ec) * (BvpB - BvpA) * 0.5;
      var AvpB = BvpB + (1 + ec) * (BvpA - BvpB) * 0.5;
      // 충돌에 영향을 주지 않는 수직 성분은 충돌후에도 변하지 않음
      var AvnA = BvnA;
      var AvnB = BvnB;
      // 공의 수평 및 수직성분으로부터 공의 x 및 y축 속도를 계산
      coll[i].vx = AvpA * unitDiffX - BvnA*unitDiffY;
      coll[i].vy = AvpA * unitDiffY + BvnA*unitDiffX;
      coll[j].vx = AvpB * unitDiffX - BvnB*unitDiffY;
      coll[j].vy = AvpB * unitDiffY + BvnB*unitDiffX;
   }
}
// end of script
```

② 자유낙하운동 및 운동량 보존 법칙 프로그램

전개부분에 적용된 자유낙하 및 운동량 보존 법칙에 따른 운동 역시 <①운동량 보존 및 충돌 검출 프로그램>에서의 충돌 검출 같은 맥락이라 볼 수 있다. 단지 공들끼리 충돌하는 것이 아니라, 공간에 떠 있는 정사각형의 평면과의 충돌이라는 점이 다를 뿐이다.

공들에게 중력가속도를 적용하여 외부로부터의 작용이 없을 경우 아래로 낙하하며, 낙하 도중 평면과 맞닿을 경우 z 벡터의 방향이 순간적

으로 역방향이 되어 튀어 오르게 된다. z 벡터는 지속적으로 중력가속도의 영향을 받기 때문에 공이 튀어 오르는 속도는 서서히 감소하다가 정점을 지나게 되면 다시 낙하하게 된다.

이러한 과정을 계속 반복하게 되며, 공이 평면 밖으로 튀어나가게 되면 투명도를 조절하여 서서히 사라지도록 만들었다.

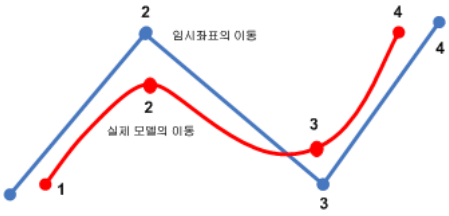
// [프로그램 5] k_grav.js

```
case 1:
   if(this.x > Xmx_g + rad_g || this.x < Xmn_g - rad_g) this.status = 2;
   this.x += this.spd_x;
   if(Math.abs(this.y - Ymn_g + rad_g) > 0.005) {
      if(this.y + this.spd_y < Ymn_g + rad_g) {
         this.spd_y = -1 * this.spd_y;
         this.y = Ymn_g + rad_g;
         // 평면과의 충돌 검출 및 벡터 역전
      } else this.y += this.spd_y;
   } else this.y = Ymn_g + rad_g;
   if(this.z > Zmx_g + rad_g || this.z < Zmn_g - rad_g) this.status = 2;
   this.z += this.spd_z;
   this.spd_x *= iner;
   // 관성의 작용
   this.spd_y -= grav;
   // 중력가속도의 작용
   this.spd_z *= iner;
   this.a += spd_g * (1.0 - this.a);
   break;
   case 2:
      if(this.y <= Ymn_g) this.status = 0;
      else {
         this.x += this.spd_x;
         this.y += this.spd_y;
         this.z += this.spd_z;
         this.spd_x *= iner;
         // 관성의 작용
         this.spd_y -= grav;
         // 중력가속도의 작용
         this.spd_z *= iner;
```

```
            this.a += spd_g * (0.0 - this.a);
        }
    break;
// end of script
```

③ 곡선 운동 프로그램

발전부분에 사용된 부드러운 공의 움직임은 2차 추적이라는 형태에 기반을 두고 있다. 화면에는 나타나지 않는 A라는 좌표가 선형적으로 움직이고, 실제 3D 모델의 좌표가 되는 B가 A를 향해 부드럽게 다가가는 것이 그 기본 개념이다.



[그림 10] 3D 모델의 부드러운 이동을 위한 기본 이론

따라서 이를 구현하기 위해서는 1개의 3D 모델 당 2개의 X, Y, Z 좌표가 필요하게 된다. 1개의 좌표는 화면에는 나타나지 않는 직선적인 좌표 이동을 위해 사용되며, 나머지 1개의 좌표는 선행하는 좌표를 천천히 따라가면 부드러운 곡선 운동이 이루어진다.

```
// [프로그램 6] k_free.js
function move() {
   if(Math.random() < 0.25) {
      this.tx = (Math.random() - 0.5) * 2 * Xmx_f;
      this.ty = (Math.random() - 0.5) * 2 * Ymx_f;
      this.tz = (Math.random() - 0.5) * 2 * Zmx_f;
   }
```

```
  // gx, gy, gz는 임시좌표, x, y, z는 실제 모델의 좌표
  this.gx += spd_f1 * (this.tx - this.gx);
  this.gy += spd_f1 * (this.ty - this.gy);
  this.gz += spd_f1 * (this.tz - this.gz);
  this.x += spd_f2 * (this.gx - this.x);
  this.y += spd_f2 * (this.gy - this.y);
  this.z += spd_f2 * (this.gz - this.z);
  // 모델의 컬러 변화: Alpha, Red, Green, Blue
  this.a += 0.05 * (this.ta - this.a);
  this.r += 0.05 * (this.tr - this.r);
  this.g += 0.05 * (this.tg - this.g);
  this.b += 0.05 * (this.tb - this.b);
}
// end of script
```

앞서 이야기 했듯이 이러한 계산을 Max의 객체만을 사용하여 수행하는 것은 불가능한 일이며, 다수 데이터의 임시 저장 공간을 필요로 하는 계산의 경우 스크립트 프로그래밍이 아니면 해낼 수 없다.

④ 무중력 공간 내에서의 비행 운동 프로그램

마지막으로 절정 부분에서 사용되었던 비행 운동 프로그램에 대해 설명하고자 한다. 이 작품에서 사용된 공의 개수는 최대 12개였기 때문에 이 부분에서는 일종의 눈속임을 사용하여야 했다. 공들이 비행할 수 있는 공간에 한계를 두고, 그 한계를 벗어날 경우 다시 공을 반대쪽 지점에 등장시키는 방법을 사용했다.

```
// [프로그램 7] k_fall.js
// X축 상의 최소 값에 도달하면 반대편(x축의 최대)에서 나타나도록 설정
  if(this.x < Xmn_d) {
    this.x = Xmx_d;
    this.y = (Math.random() - 0.5) * 2 * Ymx_d;
```

```
      this.z = (Math.random() - 0.5) * 2 * Zmx_d;
      this.a = Math.random() * 0.7 + 0.3;
      this.r = Math.random() * 0.3 + 0.3;
      this.g = Math.random() * 0.3 + 0.3;
      this.b = Math.random() * 0.3 + 0.3;
   }
// x축 상의 최대 값에 도달하면 반대편(x축의 최소)에서 나타나도록 설정
else if (this.x > Xmx_d) {
      this.x = Xmn_d;
      this.y = (Math.random() - 0.5) * 2 * Ymx_d;
      this.z = (Math.random() - 0.5) * 2 * Zmx_d;
      this.a = Math.random() * 0.7 + 0.3;
      this.r = Math.random() * 0.5 + 0.3;
      this.g = Math.random() * 0.5 + 0.3;
      this.b = Math.random() * 0.5 + 0.3;
   }
// end of script
```

그리고 사용자의 입력에 따라 공들의 비행 방향이 바뀌면 그에 따라서 공들이 현재의 비행 방향으로부터 새 비행 방향으로 천천히 바뀌어 가게 된다. 이 점이 보는 사람으로 하여금 공간을 빠른 속도로 부유하고 있는 것 같은 느낌을 주게 된다.

3. VST[23]시스템을 이용한 테이프음악의 제작

앞서 이야기 했듯이, 3D 컴퓨터그래픽의 원활한 처리를 위해 실시간 고차원적인 음합성 처리는 배제해야 했다. 그렇기 때문에 작품의 음악적인 완성도를 위해서 테이프음악을 만드는 데 다양하고 섬세한 처리

23) 독일의 음악 소프트웨어 제작사인 Steinberg에 의해 1996년에 공개된 기술. 가상 스튜디오 기술(Virtual Studio Technology)라 명명된 이 기술은, Cubase SX, Nuendo등의 소프트웨어의 기반이며, 2006년 현재 가장 대중화된 컴퓨터 음악 기술 중 하나이다.

가 필요했다.

VST시스템은 독일 Steinberg[24]사가 개발한 대표적인 디지털 오디오 워크스테이션(DAW, digital audio workstation)[25]인 Cubase SX나 Nuendo를 지칭하는 말이다. 또한, 이 작품의 테이프음악을 만들기 위해서 사용된 소프트웨어는 Nuendo이다.

1) DAW와 가상악기[26]를 사용한 테이프음악의 제작

작품에 사용된 영상의 전체적인 분위기는 매우 추상적이고 비현실적이며 단순하지만, 그러한 영상 속에 등장하는 3D 모델들의 움직임은 매우 현실적이고 구체적이다. 이러한 외형과 내용의 모순을 적절히 표현하기 위해 몇 가지 음악적 선택을 해야 했다.

첫째, 피아노를 악기로 선택했다. 현실적이고 구체적인 느낌을 그 어떤 악기보다도 잘 표현할 수 있다는 의도에서 기인한 선택이었다. 가장 많은 사람들이 손쉽게 접할 수 있기 때문에, 그리고 많은 음악들에서 널리 사용되고 있기 때문에, 대중에게 현실감을 전달하기 쉽다고 생각했다.

Nuendo에서 피아노 음색의 연주 및 녹음을 위해 사용된 가상악기는 Native Instruments[27]사의 Akoustik Piano이다.

---

24) http://www.steinberg.net/

25) 디지털 오디오 데이터의 편집 및 재생, 합성 등을 위해 사용하는 소프트웨어

26) virtual instruments. 컴퓨터의 연산능력을 이용하여 실시간으로 음합성 및 재생을 할 수 있도록 제작된 프로그램. 일반적인 전자 악기와 비슷한 구조를 가지나 범용 컴퓨터를 그 기반으로 한다는 점이 다르다.

27) http://www.native-instruments.com/

[그림 11] Akoustik Piano 가상악기

둘째, 리디아선법(Lydian mode)[28]을 사용하여 작곡하였다. 리디아 선법은 그 구성음 안에 어보이드 노트(avoid note)[29]를 가지지 않은 유일한 선법이며, 그러한 연유로 신비스럽고 모호한 감정을 표현하기 위해 사용되는 경우가 있다. 이 작품에서도 그러한 감정을 적극적으로 표현하기 위해 사용되었으며, 특히 선법의 캐릭터 노트(character note)[30]인 반음이 올라간 4번째 음을 멜로디에 빈번히 사용하여 신비감과 애매모호함을 증폭시켰다.

28) 4음이 반음 올라간 장조의 음계
29) 선법의 구성음 들중 불협화적인 성격으로 인해 사용을 피해야 하는 음
30) 선법의 구성음 들중 해당 선법의 특징을 강하게 나타내는 음

2) 그래뉼러 합성(granular synthesis)을 위한 플러그인(plug-in)[31]의 사용

작곡 과정에서 음악적으로 리디아선법을 사용한 것만으로는 비현실적이고 추상적인 느낌을 표현하기에 부족하다고 생각했다. 그리하여 음악적인 선택에 이은 음향적인 선택을 해야 했는데, 그 선택은 녹음된 피아노 소리에 보다 추상적인 느낌을 추가하기 위하여 플러그인 처리를 하는 것이었다.

일반적으로 플러그인 소프트웨어는 플러그인의 사용이 가능한 호스트 소프트웨어(host application)에서 플러그인 소프트웨어를 불러오는(load)하는 형태로 사용된다. 또한, 플러그인 소프트웨어는 호스트 소프트웨어 없이는 사용할 수 없다.

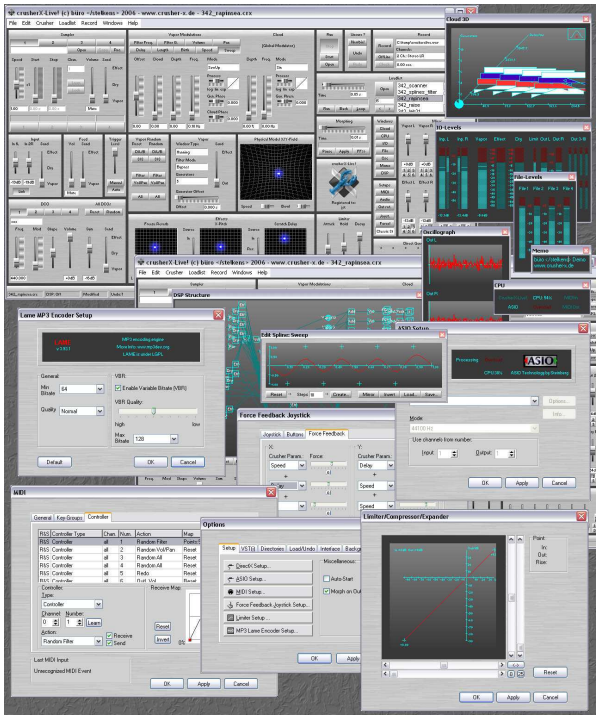
플러그인으로 사용된 크러셔엑스-라이브(crusherX-live)[32]는 고차원의 「그래뉼러」 합성 기법을 사용할 수 있도록 제작된 소프트웨어이다. 이미 기록된 파일이 있어야만 처리가 가능했던 기존의 「그래뉼러」 합성과는 달리, 실시간으로 오디오 트랙으로부터 재생되는 소리를 받아 「그래뉼러」 처리를 할 수 있다는 장점이 있다.

「그래뉼러」 합성 기법은 샘플링(sampling) 되어 저장된 디지털 오디오 데이터를 그레인(grain)이라는 매우 작은 단위의 조각으로 쪼개어, 그 순서를 바꾸거나 반복시키는 등의 방법을 통해 소리를 완전히 다른 양상으로 바꿀 수 있는 음합성 방법이다.

---

31) 어떤 프로그램에 새 기능을 추가하기 위해 끼워 넣는 부가 프로그램. 자체적으로는 실행능력은 없지만 특정한 프로그램 속에서 함께 실행되어 기능을 발휘 한다.

32) http://www.crusher-x.de/



[그림 12] CrusherX-Live 플러그인 소프트웨어

Korg[33]사의 「마이크로컨트롤」(microKONTROL)을 사용한 실시간 제어를 통해 미리 녹음된 피아노 소리를 직접 들으며 「그래뉼러」 처리를 수행하였으며, 그 결과로 작품의 테이프음악이 재생될 때에도 사람이 직접 조작한 듯한 느낌을 줄 수 있었다.

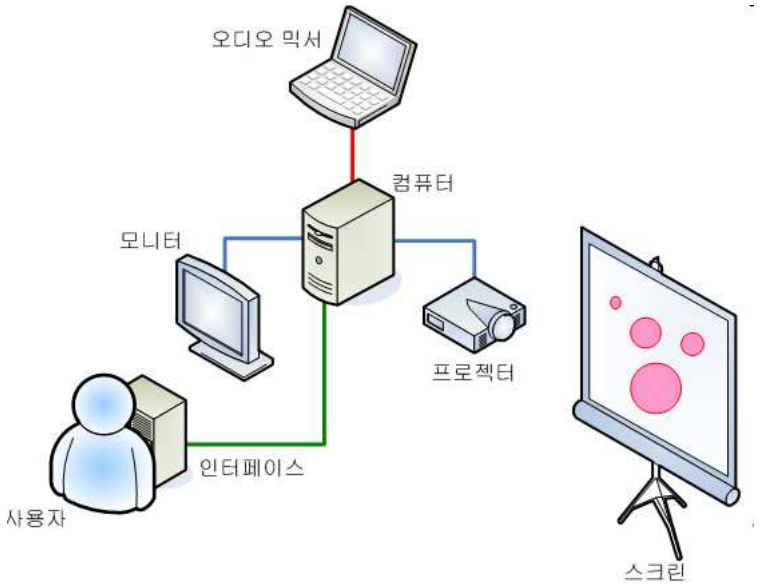
33) http://www.korg.com/

[그림 13] Korg의 「마이크로컨트롤」

4. 작품 『NATUM』의 실연

1) 연주를 위한 시스템의 전체적 구조

다음은 작품의 실제 연주를 위해 구성된 시스템의 개요도이다.



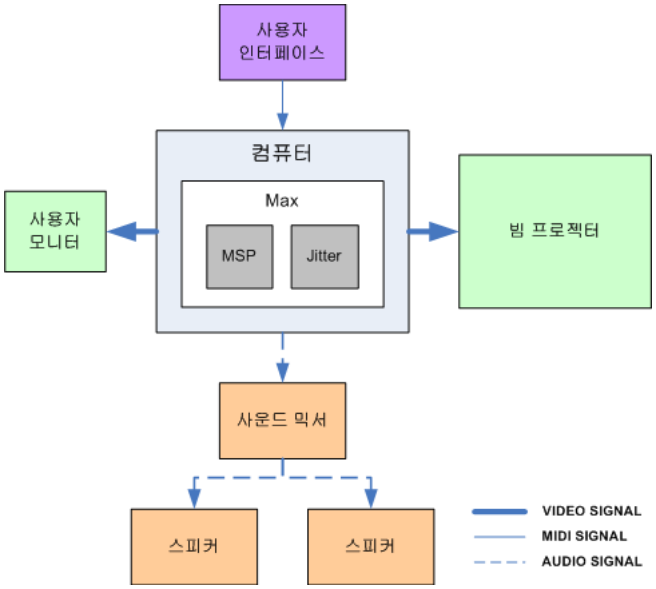
[그림 14] 연주 시스템의 개요도

[그림 14]에서 녹색 신호는 「디지타이저」에서 컴퓨터로 전송되는 사용자의 입력에 해당한다. 청색 신호는 모두 컴퓨터에서 나오는 영상 신호에 해당하며 이들은 각각 LCD 모니터와 프로젝터(projector)[34]로 전송된다. 적색 신호는 음성 신호에 해당하며, MSP로부터 재생되는 음향

34) 슬라이드·투명지(透明紙) 위의 사진·그림·문자 등을 렌즈를 통해서 스크린 위에 확대 투영하여 많은 사람에게 동시에 보여 주는 광학장치

을 오디오 믹서로 전달하는 경로가 된다.

LCD 모니터는 연주자를 위해 필요한 정보를 나타내며, 현재 작품의 어떤 부분을 연주하고 있으며 장면의 번호가 몇 번인지 등을 표시한다. 이와 동시에 프로젝터는 청중들을 위한 3D 컴퓨터그래픽 영상을 영사한다.



[그림 15] 연주 시스템의 신호 흐름도

이러한 영상의 전송을 위해 듀얼 모니터(dual monitor)[35] 사용이 가능한 비디오 카드(video card)가 필요했으며, 연주를 위해 사용된 컴퓨터에 장착되어 있던 GeForce[36] 비디오 카드는 듀얼 모니터 기능을 지원하기 때문에 문제는 없었다.

---

35) 하나의 비디오 어댑터로 2개의 디스플레이를 사용하는 것을 말한다.

36) 미국 엔비디아(nVIDIA)의 고속 그래픽 카드 칩 세트 시리즈를 말한다. 캐나다 ATi의 라데온(Radeon) 시리즈와 함께 게임용으로 사용 된다

2) 센서 시스템의 제작

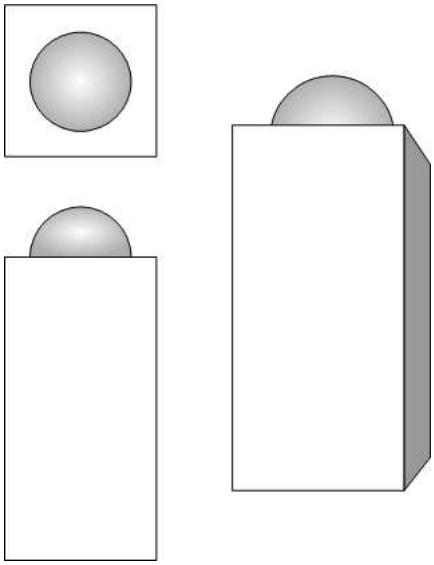
효율적인 작품의 연주 및 무대 연출을 위해 센서 시스템을 직접 제작했다. 직접 설계한 악기의 모형에 센서를 부착하고, 센서를 「블루투스」 「디지타이저」에 연결하는 것으로 악기를 완성했다. 악기 모형의 구체적인 설계도는 이 논문의 첨부된 부록에서 다루고 있다.

사용자의 입력 감지를 위해 사용된 것은 FSR(force sensing resistor)[37]센서였으며, 센서 키트(kit)를 이용하여 제작된 센서들은 악기 모형의 상단 반구에 부착되었다.

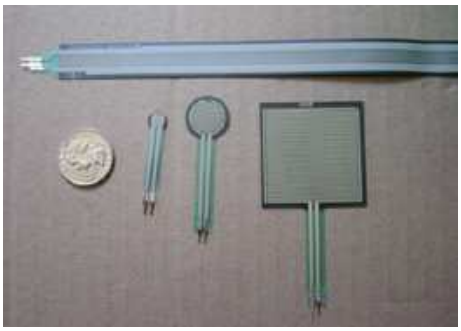
「블루투스 디지타이저」는 PC에 연결되어 윈도우 기반의 편집기로 설정 및 조작할 수 있었는데, 여기서는 보다 효율적으로 사용자 입력을 받아들이기 위해 몇 가지 설정이 필요했다.

입력은 연속적인 값의 형태가 아니라 순간적인 것이므로 Continuous 설정은 해제했으며, 미디 데이터의 끝을 알리는 End 설정을 활성화 시켜서 건반을 누르는 것과 같은 형태로 미디 신호를 전송할 수 있도록 했다.

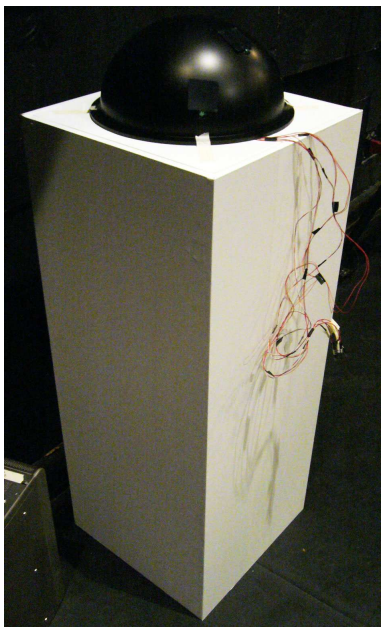
37) 센서 표면에 힘을 증가 시킬 때 감소하는 저항이 발생하는 중합체의 필름(Polymer Film) 장치



[그림 16] 모형의 구상도



[그림 17] FSR 센서 키트

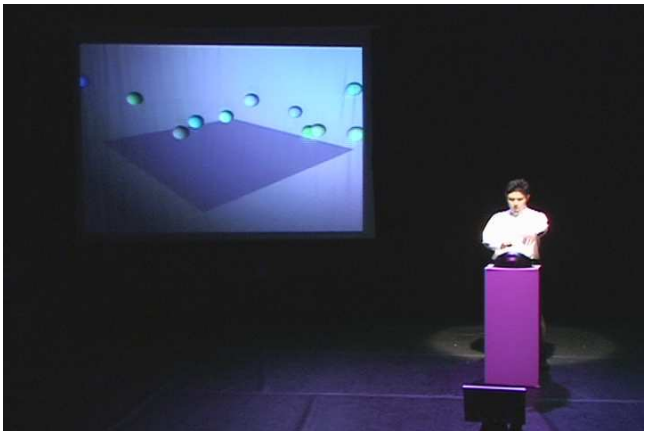


[그림 18] 완성된 악기 모형

[그림 19] I-CubeX Wi-miniDig

3) 무대 구성 및 조명 설정

무대는 최대한 단순하게 구성했다. 연주자와 악기, 빔 프로젝터가 영사하는 스크린으로 이루어졌으며, 스크린을 왼쪽, 연주자와 악기를 오른쪽에 배치하여 청중들이 「인터랙션」(interaction)[38]을 잘 관찰할 수 있도록 했다.



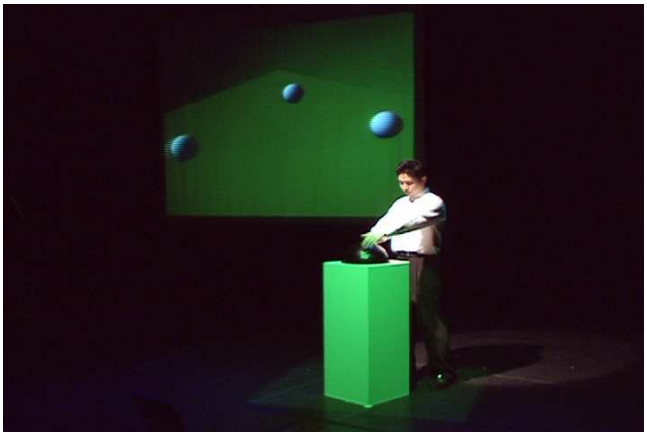
[그림 20] 무대의 구성

도입, 전개, 발전, 해결부분에서는 연주자의 머리 위에서 직접 내리비추는 핀 조명을 사용하였으며, 절정 부분에서는 무대 전체를 희미하게 비추는 조명을 사용하여 어지러운 영상과 일체감을 줄 수 있도록 했다.

악기의 모형은 흰색의 직육면체와 검은색의 반구를 결합하여 제작했으며, 흑백의 조화를 통해 스크린에 영사될 컴퓨터 그래픽과 어울리도록 했다. 또한, 흰색의 직육면체는 무대 조명의 색상에 따라 쉽게 그 색이 변화 하였으며, 이로 인해 영상과의 일체감을 얻을 수 있었다.

---

38) 상호 작용, 상호의 영향



[그림 21] 녹색 조명을 받은 무대 및 모형

[그림 22] 파란색 조명을 받은 무대 및 모형

III. 결 론

1. 문제점

작품을 처음 구상했을 때부터, 문제가 되리라 생각했었던 몇 가지 문제점들이 있었지만 그 중에서 생각만큼 크게 문제가 된 것들은 없었다. 그럼에도 불구하고 아직 부족하거나 해결해지 못한 부분들이 있었으며 그것들은 다음과 같다.

첫째로, 「인터랙티비티」의 부족함이다. 일단 음향재생의 많은 부분은 테이프음악에 의존하고 있다는 점이다. 미리 녹음하여 준비해 둔 부분이 많다는 것은 그 만큼 실시간의 상호작용성이 부족하다는 의미가 된다.

그리고 절정부분의 컴퓨터 그래픽의 경우, 사용자의 입력을 받은 뒤 실제 공의 비행 방향이 바뀌기 까지는 조금 시간이 걸리는데, 이것은 이 움직임에 사용된 공식이 가지는 특성 때문이었다. 이유는 차치하고 즉각적인 반응이 나타나지 않는다는 점에서 볼 때 「인터랙티비티」는 부족하다고 말할 수 있다.

둘째로, Jitter가 제공하는 OpenGL의 수준에 대한 문제이다. 작품의 구상 시부터 염두에 두었던 것 중의 하나가 바로 효율과 성능이다. 스크립트 프로그래밍을 사용하여 효율을 크게 올렸음에도 불구하고, 객체와 그래픽 유저 인터페이스를 사용하고 있는 높은 수준(layer)의 OpenGL 프로그래밍은 효율 면에서는 떨어질 수밖에 없다.

특히 3D 모델이 빠른 속도로 이동하는 움직임을 보일 때는 화면의 3D 모델들의 일부분이 깨져 보이는 등의 문제점이 발생했다. 그리고 일부 그래픽 카드를 탑재한 컴퓨터에서는 3D 모델들의 순서(depth)를

나타내는 데 있어서 문제가 발생했다. 공간상 뒤에 있어야 할 3D 모델들이 앞으로 나와 보이는 등의 문제점이 발견되었다.

뿐만 아니라, 텍스트를 화면에 표시하는 객체에도 문제가 있었는데 텍스트에 안티-에일리어싱(anti-aliasing)[39]이 전혀 적용되지 않았으며, 텍스트의 크기조차 제대로 조절하지 못하는 불편함이 있었다.

셋째로, 구상했던 대로 FSR 센서 키트를 사용하지 못했다.. 작품에 사용된 센서는 FSR 센서 키트 중에서도 정사각형의 넓은 면적을 갖고 있는 센서였는데, 이 센서를 악기 모형의 상단 반구에 부착하자 제대로 동작하지 않았다. 본래 평면 형태인 센서가 구면에 맞게 휘어지면서 감지 기능을 상실하게 된 것이다. 그리하여 센서와 구면을 중심의 일부분만 부착함으로써 해결하였다.

이것은 쉽게 떨어질 수 있는 문제점을 안고 있으며, 나아가 설치 형태로 작품이 제작될 경우 일반 대중들의 손에 닿으면 쉽게 떨어질 수 있다는 것을 의미하기도 한다.

넷째로, 작품의 구상 단계에서는 사용자 입력의 강도를 감지하여 영상에서의 움직임의 속도를 변화 시키도록 하였으나, FSR 센서의 특성상 다양한 강도를 얻는 것이 쉽지 않았다. 또한, 앞서 이야기 한 바 있는 보간의 속도에 영향을 미치는 계수를 정하는 작업을 충분히 하지 못했기 때문에, 시점의 이동 속도 및 3D 모델의 색상 변화의 속도가 사용자가 생각하고 입력하는 대로 이루어지지 않는다는 느낌이 들었다.

다섯째로, 3D 모델의 움직임에 적용된 물리적 운동 자체가 너무 구체적, 현실적이기 때문에 작품 전체에 나타나고 있는 추상적인 느낌에 오히려 방해가 되었다. 이러한 상황을 막기 위해, 물리적 운동을 위한 시

39) 그래픽 프로그램에서 화면의 영상을 자연스럽고 매끄럽게 표현하도록 해 주는 화면 처리 기법

스템을 구축해 놓은 뒤 그 시스템에 극단적인 변수를 넣어 현실감을 떨어뜨리는 방법이 있지만 이번 작품에서는 사용되지 않았다.

2. 결과

멀티미디어와 상호작용, 그리고 실시간이라는 세 가지 요소를 통합하기 위해 다양한 기술들이 작품 전반에 걸쳐 사용되었다. 이 기술들이 바로 이 작품의 제작을 통해 얻게 된 값진 결과라 하겠다.

첫째, Max에 스크립트 프로그래밍이 더해짐으로 해서 보다 한 차원 높은 멀티미디어 및 「인터랙션」의 실시간 처리가 가능하다는 것이다. Max/MSP/Jitter로 제공되는 객체들의 조합만으로는 불가능한 일들도 간단한 스크립트 프로그램을 추가하여 쉽게 해결할 수 있는 경우가 많았다.

둘째, 시스템이 훌륭한 컨텐츠와 「인터랙션」을 갖고 있다고 해도, 사용자와의 인터페이스가 직관적이지 못하고 접근성이 부족하다면 아무런 소용이 없다고 할 수 있다. 그런 의미에서 FSR 센서와 「디지타이저」를 조합한 악기의 제작은 좋은 경험이 될 수 있었다.

셋째, 「인터랙티브」 미디어의 「인터랙티비티」를 높이기 위해 실시간 렌더링이 가능한 애니메이션을 사용하는 것이 어떤 의미를 가지는지 깨달았다는 것이다. 실시간 렌더링이 가능한 그래픽에 실시간 연산을 통한 애니메이션이 더해지면, 정보 전달 효과는 배가 되며 사용자의 반응도 또한 높았다.

그리고 다양한 조건과 인수(parameter)를 통해 실시간으로 영상을 렌더링 할 수 있다면, 사용자가 이미 만들어진 시나리오를 따라가는 것이 아니라 매번 새로운 시나리오 그 자체를 만들 수 있는 자유도를 제공

할 수 있으며, 계속 새로운 궁금증을 사용자에게 제공할 수 있는 가능성도 갖게 된다. 때문에 OpenGL은 「인터랙티브」 미디어를 위한 훌륭한 도구라고 할 수 있다.

3. 설치작품형태에 대한 계획

전술한 바와 같이 작품 『NATUM』은 원래 설치작품형태로 기획되었다. 이 논문은 연주 형태로 부분 수정 및 변형된 작품을 중심으로 다루고 있지만, 설치작품형태의 작품도 언급해야 할 필요는 있다. 특히, 설치작품으로서 갖추어야 할 요소들은 다음과 같이 지적할 수 있다.

첫째, 사용자 인터페이스 즉, 사용자의 손에 닿는 부분들을 더욱 완전하고 견고하게 만들어야 할 필요가 있다. 응답성이 뛰어나고 부착이 용이한 센서 시스템을 사용하면 문제를 쉽게 해결할 수 있으리라 믿는다.

둘째, 더 큰 시즐감(sizzle)[40]을 주어야 한다는 것이다. 시즐감이라는 것은 광고에서 주로 사용되는 용어로 영상에 부합하는 음향 효과를 더욱 극대화 시켜, 보는 사람이 광고 대상에 대해 느끼게 되는 더 강한 감정을 의미한다.

「인터랙션」이 얼마나 효과적인가는 결국 사용자가 무언가를 조작했을 때 그에 대한 결과로 느끼게 되는 시즐감이 얼마나 강한가에 달려있다. 이는 결국 행동 심리학과 심리 음향학, 시각 디자인 등에 깊게 관련된 사항이며, 이를 잘 이해하고 적용하기 위해서는 심도 있는 학습도 반드시 수반되어야 할 것이다.

셋째, 이 작품이 실세계를 모방한 물리적 체계에 대한 학습과정이었다

40) 어떤 제품의 광고효과를 위해 그 제품의 핵심 포인트가 될 만한 소리를 활용하는 광고기법

면, 이제부터는 이 체계 기반으로 한 추상적, 비현실적인 애니메이션을 추구해야 한다는 것이다. 이는 비단 보이는 것뿐만이 아니라 들리는 것에도 해당하는 바이다.

더 추상적이며, 더 비현실적인 영상과 음향이 조화될 때, 이 작품의 기본 밑바탕인 '상징'의 의미를 한 층 두텁게 만들어 줄 것이다.

넷째, 무한 반복성을 작품에 추가해야 한다는 것이다. 연주 작품은 연주 시간을 갖고 있으며 작품의 시작과 끝이 분명히 존재한다. 하지만 설치 형태의 작품은 시작과 끝이 없으며, 작품이 철거되는 순간 까지 무한히 반복될 수 있어야 한다.

앞서 말했듯이, 이러한 이유 때문에 선형적 시나리오 시스템이 아닌, 어떠한 시나리오라도 제작할 수 있는 '체계' 그 자체를 만드는 것이 중요한 것이다.

검색어(Keyword): Max/MSP, Jitter, OpenGL, 가상악기(virtual instrument), 멀티미디어음악(multimedia-music), 인터랙티브 아트(interactive arts), 컴퓨터음악(computer-music)

E-mail: melvin@rvmusic.co.kr

참고문헌

- 신명용, 「플래시 MX 액션스크립트」, 서울:제우미디어, 2002
- Furuhata, Kszuhiro 저/하주석 역, 「Java Script 대백과 사전」, 서울:홍익미디어CNC, 2001
- Hirmes, David 외 공저/허영주 역, 「수학으로 디자인 한 플래시의 세계」, 서울:에이콘출판사, 2003
- Netscape 개발자 포럼, 「Java Script Guide/Reference」
  http://developer.netscape.com/library/documentation/communicator/jsguide4/index.htm
- Cycling '74, 「Max/MSP Tutorial」, PDF File, Version 4.5.5
- Cycling '74, 「Jitter Tutorial」, PDF File, Version 1.5.2
- OpenGL.ORG.「OpenGL Reference Blue Book」, HTML File, Version 1.4
  http://www.opengl.org/documentation/blue_book_1.0/
- Max/MSP 사용자 포럼(Hypermail archive)
  http://www.synthesisters.com/hypermail/max-msp/

Abstract

Implementing the integration of multimedia and interaction music based on Max/MSP and Jitter/OpenGL (producing interactive multimedia music - 'NATUM')

It needs hardly to be said that computer is the best tool to make human's life. Although the ultimate purpose of the developing computer technology was calculating numbers and various logical operation, the computer have been settled as outstanding artistic instruments as digital media is spreading into our life rapidly.

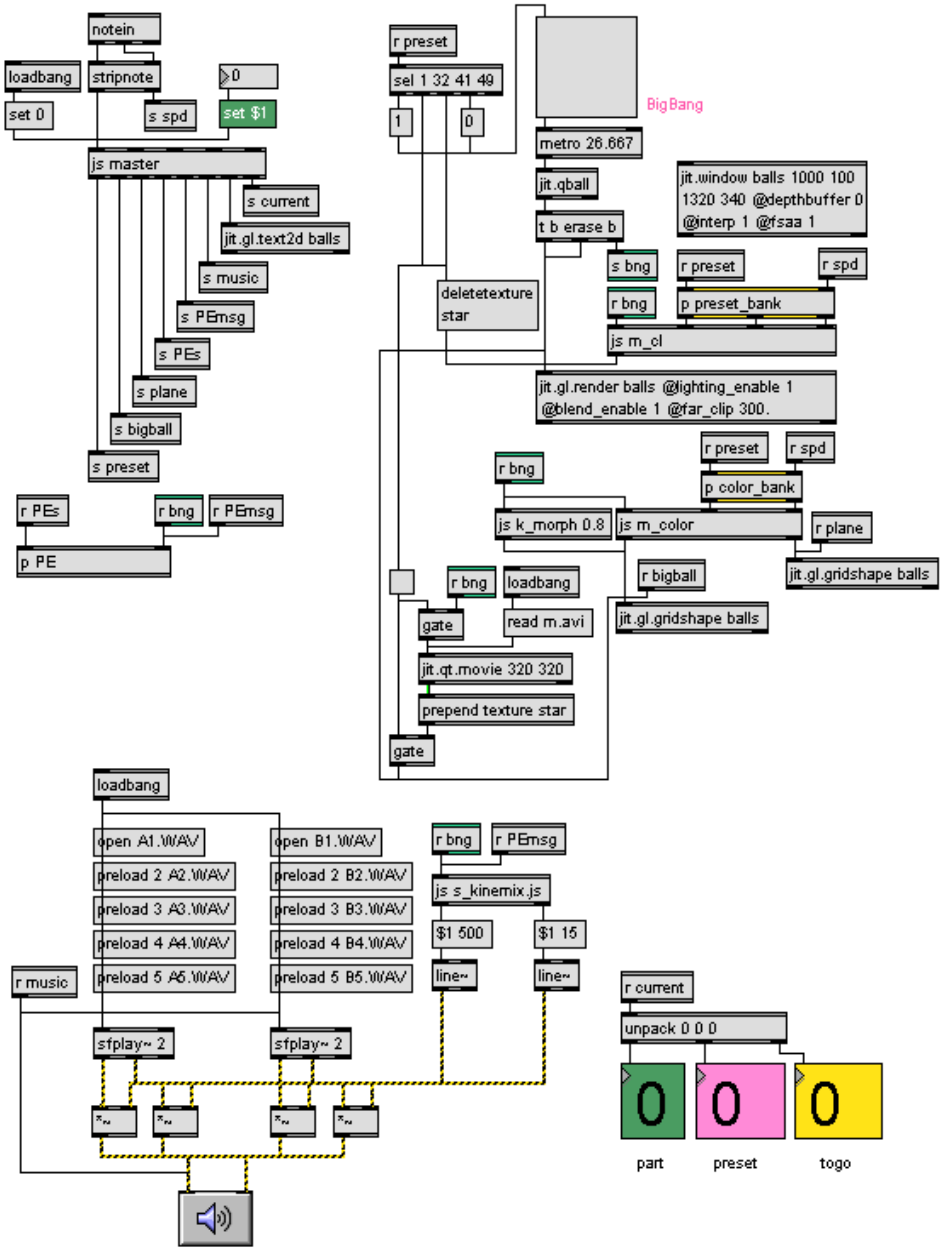
『NATUM』, the interactive multimedia artwork, makes its final objective to integrating visual, music and human based on the computer.

Technically, The main tool of this artwork is Max/MSP which is popular and widely used for multimedia programming environment. And especially, Jitter and OpenGL was used for high-level video processing.

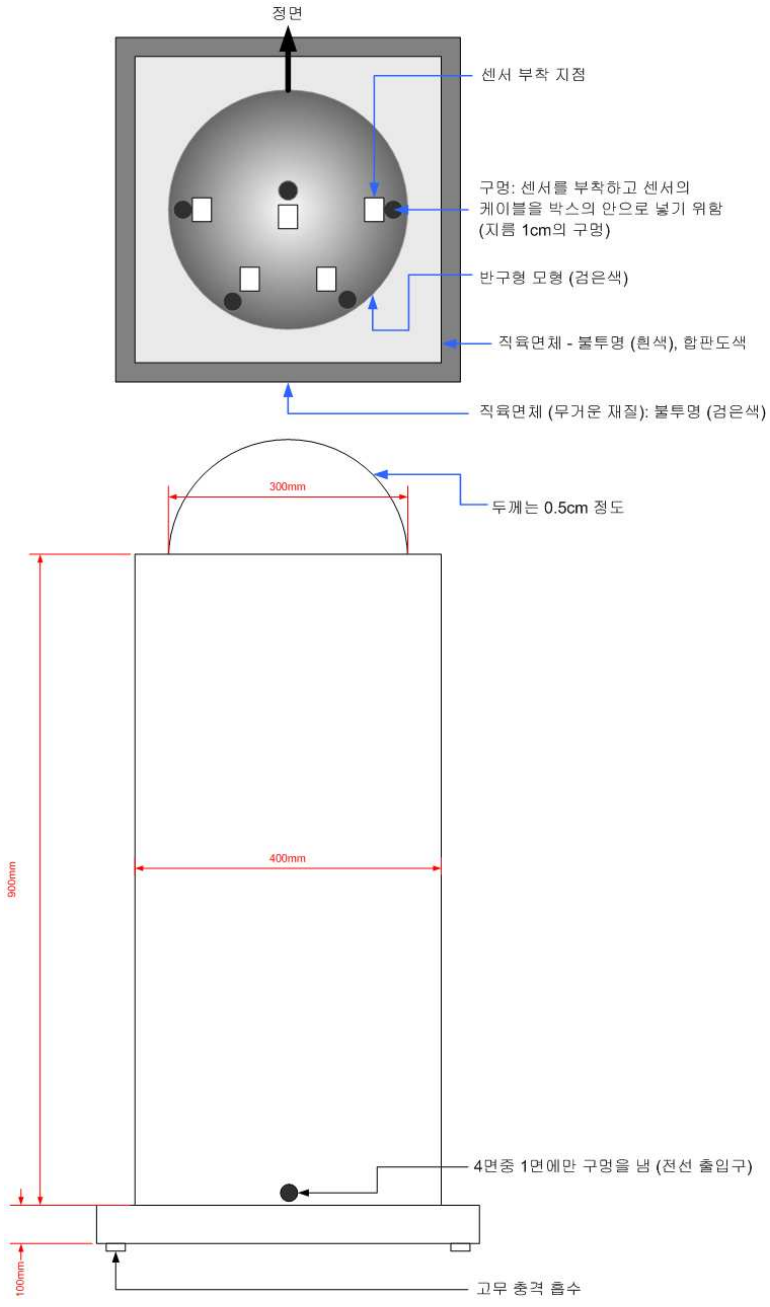
The important measure of this artwork is the interaction using 3-dimensional computer graphic and synthesized sound. And the key point is using 3-D graphic animation based on kinetic movement to make audience well understood interaction. JavaScript feature which is integrated with Max/MSP makes kinetic movement possible.

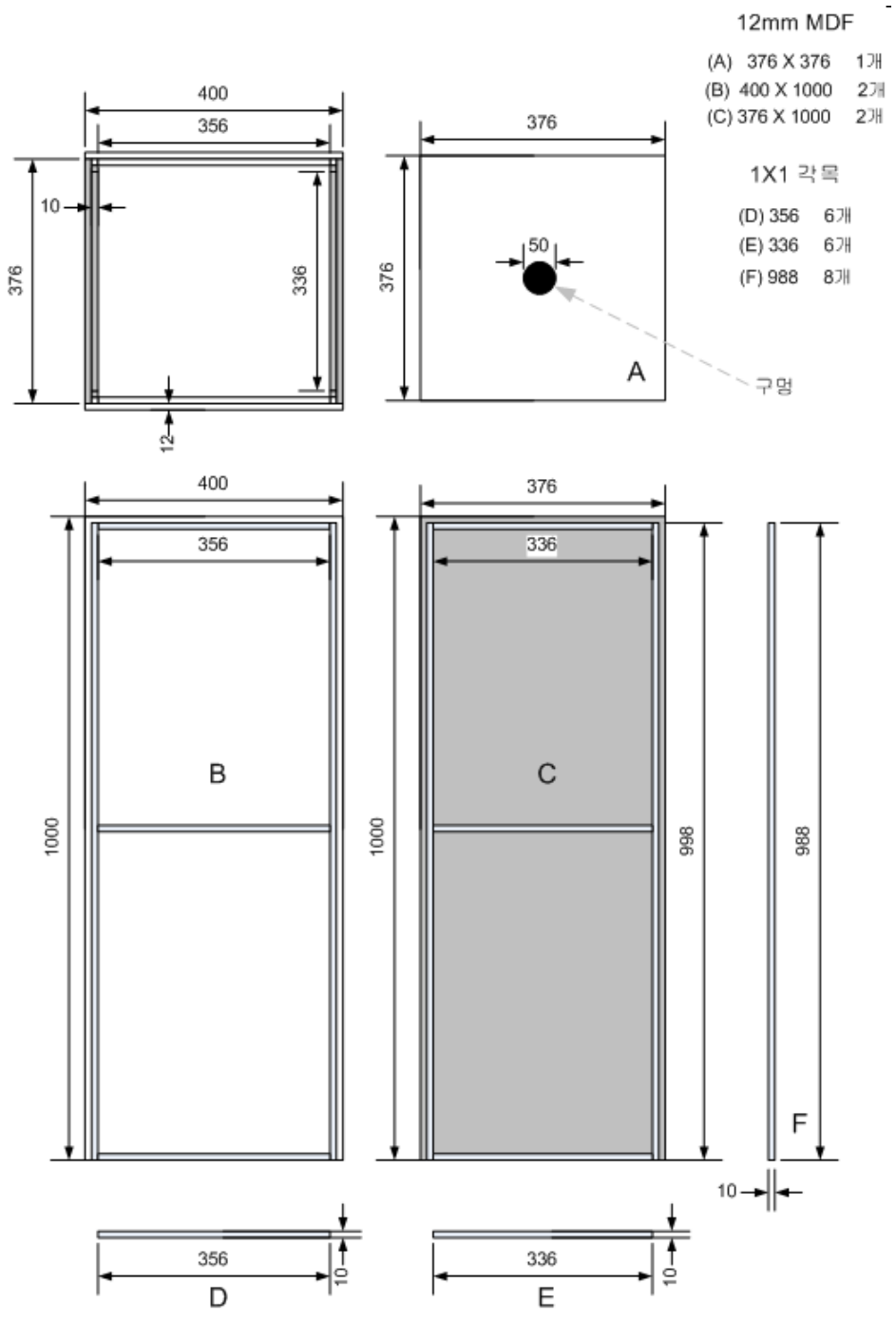
It is the great outcome to obtain information about the efficiency of the interactivity in multimedia arts using 3-D graphic animation.

부록 1: Max/MSP의 패치 구성도



부록 2: 악기 모형의 설계도




12mm MDF
(A) 376 X 376 1개
(B) 400 X 1000 2개
(C) 376 X 1000 2개
1X1 각목
(D) 356 6개
(E) 336 6개
(F) 988 8개
400
356
376
10
336
12
376
50
A
구멍
400
356
B
1000
376
336
C
1000
998
988
F
10
356
10
D
336
10
E

부록 3: 첨부 CD의 내용 설명

① NATUM.avi
공연실황 녹화 동영상

② NATUM.mxb
작품 『NATUM』의 Max/MSP 패치

③ A1.wav, A2.wav, A3.wav, A4.wav, A5.wav
5개의 부분을 위한 테이프음악 A

④ B1.wav, B2.wav, B3.wav, B4.wav, B5.wav
5개의 부분을 위한 테이프음악 B

⑤ k_coll.js
충돌 검출 및 운동량 보존을 위한 프로그램 코드

⑥ k_fall.js
무중력 공간의 비행을 위한 프로그램 코드

⑦ k_free.js
무중력 공간의 부드러운 이동을 위한 프로그램 코드

⑧ k_grav.js
자유낙하와 운동력 보존을 위한 프로그램 코드

⑨ k_morph.js

3D 모델의 자연스러운 확장 및 축소를 위한 프로그램 코드

⑩ m_cl.js

3D 공간의 시점 이동과 조명의 위치 변화를 위한 프로그램 코드

⑪ m_color.js

3D 모델들의 자연스러운 색상 변화를 위한 프로그램 코드

⑫ master.js

작품 전체의 진행 및 사용자의 입력을 처리하는 프로그램 코드

⑬ m.avi

3D 모델의 텍스쳐 매핑을 위해 필요한 동영상 파일