Journal of Korea Game Society 2015 Jun; 15(3): 189-200
http://dx.doi.org/10.7583/JKGS.2015.15.3.189

컴퓨터 그래픽스

WebGL을 이용한 블록 기반 컴퓨터 그래픽스 교육용 소프트웨어 모델

편해걸, 박진호
숭실대학교 글로벌미디어학부
phg2491@naver.com, c2alpha@ssu.ac.kr

A Block-based Computer Graphics Educational Software Model using WebGL

Hae-Gul Pyun, Jinho Park,
Global School of Media, Soongsil University

요 약

오늘날 많은 IT 관련 분야에서 컴퓨터 그래픽스 기술이 사용되고 있다. 더욱이 3D 프린터, Head Mount Display, VR & AR 등 컴퓨터 그래픽스와 밀접하게 관련된 분야에 대한 수요가 급증하고 있다. 앞으로 컴퓨터 그래픽스 분야는 더욱 전문화되고 이에 따른 인력의 수요도 증가할 것이다. 그러나 그래픽스 분야가 수학적 배경지식을 많이 요구하기 때문에 접근성이 낮고, 수요에 비해 이를 전공한 사람과 전문가의 숫자가 적다. 만약 그래픽스 프로그래밍을 쉽게 배울 수 있는 환경을 제공하다면, 컴퓨터 그래픽스 분야 인력 양성에 도움이 될 것이다. 따라서 이 논문에서 그래픽스 이론을 분석하여 초심자도 체계적이고 쉽게 배울 수 있는 교육용 소프트웨어 모델을 제시한다. 웹과 블록을 이용한 설계를 통해 접근성과 직관성을 높이고, 이론적인 내용을 중점적으로 학습할 수 있는 환경을 구축하는 방법을 제안한다.

ABSTRACT

These days computer graphics technology has been applied in diverse IT fields. Needs for computer graphics such as 3D Printer, Head Mount Display, VR & AR are growing rapidly. Computer graphics will be more specialized and demanding for graphics specialists will be also increased. However, serious mathematical background obstructs people to learning computer graphics. An efficient computer graphics learning system would be helpful for graphics experts training. By analyzing the graphics theory, we propose an educational software system with that students can effectively learn computer graphics. Our system focuses on theoretical objects of computer graphics and enhances accessibility and intuition using web and blocks.

Keywords : Computer Graphics(컴퓨터 그래픽스), WebGL(웹 지엘), E-learning(이러닝), Block Based (블록 기반), Interactive(상호 작용)

Received: May. 14. 2015 Accepted: Jun. 15. 2015
Corresponding Author: Jinho Park,(Soongsil University)
E-mail: c2alpha@ssu.ac.kr

ISSN: 1598-4540 / eISSN: 2287-8211

1. 서 론

프로그래밍을 배울 때 하나의 언어 원리를 습득하면 다른 언어를 배우기 쉽다. 마찬가지로 그래픽스 프로그래밍의 원리를 보다 쉽게 이해할 수 있으면, 그래픽스 분야에 대해서 전반적으로 이해하는데 큰 도움이 된다. 소프트웨어 교육에 대한 관심이 높아지고 중요해지는 만큼 중 고등학생들이 컴퓨터 그래픽스 분야에도 쉽게 접근할 수 있도록 하고자 한다. 따라서 이 논문에서 그래픽스 이론을 분석하여 이를 블록을 이용해 쉽게 배울 수 있는 교육용 소프트웨어 모델을 제시한다. 블록 조합을 통해 사용자들이 쉽게 컴퓨터 그래픽스 이론을 학습하고, 이를 응용하여 모델링까지 가능하도록 구현한다.

현재 세계는 소프트웨어 교육 열풍이 불고 있다. 2014년 핀란드 SW 학교 전국 확대, 2014년 9월 영국 G20 국가 중 최초로 SW 교육 정규 필수 과목 지정, 미국도 작년 가을부터 컴퓨터 프로그래밍 수업을 정규 교과과정으로 편성했다. 한국은 올해 중학생부터 SW 의무 교육이 시행되며, 2018년에는 정식 교과목으로 채택 될 예정이다. 정부에서도 미래창조과학부가 주가 되어 소프트웨어 중심사회 실현을 목표로 SW 전문 인력을 양성한다고 발표했다. 모든 것이 컴퓨터 기반으로 돌아가는 현대 사회에서, SW는 이제 제2의 세계 공용어가 되어 가고 있다. 따라서 전문 개발자가 아닌 사람들을 대상으로 프로그래밍에 대한 접근성을 높히고, 쉽게 코딩을 가르치기 위하여, 각종 프로그래밍 교육 소프트웨어가 등장하고 있다. 따라서 앞으로 프로그래밍 교육용 소프트웨어 시장이 급속도로 성장할 것이다.

현재 많은 컴퓨터 분야에서 각광받고 있는 것이 컴퓨터 그래픽스이다. 조금만 유심히 살펴보면 우리 일상생활의 많은 부분에서 사용되고 있는 것을 볼 수 있다. 과학, 공학, 의학, 경영, 산업 ,예술, 오락, 광고 등 많은 분야에서 컴퓨터 그래픽스 기술이 활용되고 있다. 최근에는 3D Printer와 모델링, Head Mount Display 오큘러스, Google Glass에 쓰이는 증강 현실 등 그 필요성은 점점 부각되고 있다.

그러나 컴퓨터 그래픽스 프로그래밍은 3차원 좌표계, 행렬, 벡터 등 수학적 계산에 기초하고 있으므로 이를 배우기 위해선 수학적 지식이 필요하다. 뿐만 아니라 공간지각능력도 요구하기 때문에 프로그래밍을 웬만큼 할 줄 아는 사람도 컴퓨터 그래픽스를 배우기가 쉽지가 않다. 따라서 그래픽스 프로그래밍을 쉽게 배울 수 있는 환경을 제공하면, 컴퓨터 그래픽스 분야 인력 양성에 도움이 될 것이다[1].

2. 관련 연구

GUI기반의 프로그래밍 교육 시스템을 제안해고자 했던 몇몇 시도들이 있었다. 대표적으로 미국 MIT Media Lab에서 아이들에게 그래픽 환경을 이용하여 컴퓨터 프로그래밍에 관한 경험을 쌓게 하기 위해 만들어진 스크래치(Scratch)[2]가 있다.

스크래치는 스퀵(Squeak)을 기반으로 스몰토크(Smalltalk)라는 언어로 작성되었는데, C나 C++와 달리 블록을 드래그하여 탑을 쌓는 것처럼 프로그래밍 하여 개념을 쉽게 학습 할 수 있다. 스크래치와 같은 교육용 블록기반 프로그래밍 방식의 특징은 직관적인 언어로 초등학생들도 보다 쉽게 프로그래밍을 이해할 수 있으며, 학습자의 학습 몰입(Flow)에 긍정적인 영향을 미친다[3,4,5].

최근 웹 기반의 교육용 프로그래밍 시스템들이 개발되었다. 코드카데미(Codecademy)는 코딩을 배우고 싶은 학생들이 온라인을 통해 강의를 배울 수 있다. 강의 각 단계마다 목표가 존재하고, 메시지 지시대로 TextBox에 코딩을 한다. 코딩 후에는 서버에 제출하여 맞았는지 확인할 수 있다. 다른 서비스로는 Code.org가 존재한다. 서비스 취지는 코드카데미와 같으나 스크래치와 같은 블록 코딩 방식을 채용하여 좀 더 낮은 연령층[6]을 타겟으로

잡고 있다.

앞에서 설명한 프로그래밍 학습 시스템들은 우리가 제시하는 모델과 차이가 존재한다. 전자의 모델들은 언어의 학습이라는 분야에 초점을 맞추고 있는 반면 우리의 모델은 컴퓨터 그래픽스 분야의 이론을 학습하는데 초점을 맞추고 있다. 프로그래밍 언어의 사용법 습득과 컴퓨터 그래픽스 이론의 학습은 전혀 다른 분야이다. 그밖에 컴퓨터 그래픽스라는 분야가 같은 특성 때문에 생기는 구조적, 기술적 차이점이 존재한다[7]. 프로그래밍 언어는 문법을 익히는 반면 컴퓨터 그래픽스는 오브젝트, 카메라, 빛 등을 구현하고 다루는 이론을 다루기 때문에 이러한 것들을 학습하는데 초점을 맞춘다.

3. 우리의 연구

3.1 개요

우리는 언어의 사용법이 아닌 컴퓨터 그래픽스의 이론을 학습하는 것에 초점을 맞추고 있다. 따라서 반복문이나 조건문의 사용 방법 등은 배제하여 이론에 집중할 수 있도록 하였다. 그리고 블록 조합을 베이스로 만들어 코딩 시 발생하는 구문 에러(Syntex Error), 컴파일 에러(Compile Error) 등을 사용자가 신경 쓰지 않아도 되도록 설계 한다. 또 쉬운 내용부터 어려운 내용까지 체계적이고 단계적으로 배울 수 있는 프로세스를 제공하여 순차적으로 실력을 기를 수 있게 한다. 마지막으로 학습 후 블록을 이용하여 자유롭게 모델링을 가능하게 하는 콘텐츠를 제공하여 그래픽스 프로그래밍에 대한 흥미를 유도한다.

3.2 핵심 모델

우리가 제시하는 컴퓨터 그래픽스 교육용 소프트웨어 모델의 핵심은 쉽게 그래픽스의 개념을 이해하는 것이다. 따라서 이를 충족시키기 위하여 구현한 기능이 크게 네 가지로 웹과 블록 기반 프로그래밍, 튜토리얼 시스템, 판정 시스템이 있다. 사용자는 웹 페이지에 접속하여 그래픽스 개념에 관련된 튜토리얼을 진행한다. 튜토리얼은 그래픽스의 중요 이론에 따라 크게 3개의 챕터로 나뉘며 각각의 챕터마다 달성해야 할 목표가 존재한다. 목표를 달성하면 이를 판정하여, 올바른 답안을 도출하면 다음 챕터로 넘어가는 구조이다.

3.2.1 웹 기반

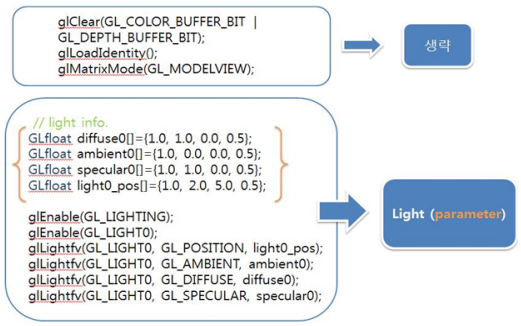
컴퓨터 그래픽스를 처음 접하는 많은 사람들이 컴퓨터 산업 표준 응용 프로그램 인터페이스(API)인 OpenGL을 기반으로 학습한다. 그러나 OpenGL을 사용하려면 통합 개발 환경(IDE)에 OpenGL을 링크 시켜야하기 때문에, 접근성과 활용도가 상당히 낮아진다. 개념을 익히는 것을 목표로 하고 있기 때문에 복잡한 소프트웨어 설치 환경은 피해야 한다. 따라서 언제 어디서나 접근할 수 있고, 별도의 소프트웨어의 설치가 필요하지 않은 웹 브라우저 기반 모델을 제시한다[8].

3.2.2 블록 기반 프로그래밍

컴퓨터 그래픽스를 처음 접하는 사람들은 복잡한 라이브러리 구조 때문에 배우는 것을 어려워한다. 그러나 우리의 모델에서는 코드 타이핑을 통한 프로그래밍 학습이 아니라 핵심 함수를 블록에 매핑하여 이를 조합하여 학습을 할 수 있도록 한다. 다시 말해, 각 함수의 기능에 초점을 맞춰 보다 쉽게 개념을 익힐 수 있도록 한다.

따라서 OpenGL에서 사용되는 API 함수들을 블록으로 맵핑한다. 블록 조합을 통해 실시간으로 랜더링 결과를 확인하면서 각각의 블록(함수)이 어떠한 기능을 하는지 학습하도록 돕는다. 그러나 하나의 오브젝트를 구현하는데 사용되는 함수들을 모두 맵핑하면, 블록의 숫자가 많아지고 구조가 복잡해진다. 따라서 사전에 기본적인 블록과 함수 설명을 첨부하고, 핵심 개념을 이해할 수 있는 최소한의

블록만 사용하여 사용자가 직관적으로 이해할 수 있도록 한다.



[Fig. 1] Block Concept

예를 들면 [Fig. 1]은 OpenGL에서 조명을 만들 때 필요한 함수들이다. 광원을 만들 때 glLightfv 함수를 사용하여 일일이 position, ambient, diffuse, specular 값들을 넣어주어야 하는데 이런 반복적인 작업들을 줄이고 좀 더 간결하게 표현하기 위해서 블록을 사용한다. 그리고 glClear, glLoadIdentity, glMatrixMode 같이 초기화 함수들은 텍스트로 개념을 설명하고, 블록 조합에선 제외 시켜서, 조명의 개념을 이해하는데 포커스를 맞추도록 한다.

각 블록마다 고유의 파라미터 값을 가지고 있으며, 사용자는 이 블록들을 조합하여 오브젝트를 만들거나 조작 할 수 있다. 기본적으론 블록을 특정한 공간에 Drag & Drop 방식으로 배치시켜서 조합을 할 수 있게 한다. 또 조합이 될 때 마다 결과 Viewport에 Rendering 하여 사용자가 결과를 실시간으로 확인할 수 있도록 한다.

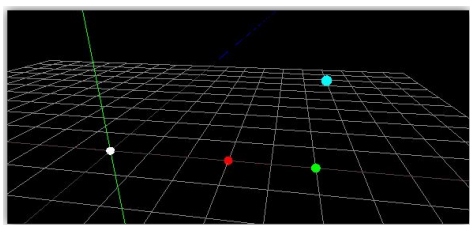
3.2.3 튜토리얼 시스템

어려운 그래픽스 개념을 쉽게 이해할 수 있도록 만드는 것에 포커스를 맞췄다. 튜토리얼 시작 시, 먼저 해당 챕터와 관련된 지식을 간략하게 설명해 주는 페이지를 보여준다. 그 다음 해당 개념을 이해하면 만들 수 있는 결과를 보여주고, 블록을 조합하여 똑같은 결과를 낼 수 있도록 유도한다. 결과적으로 사용자가 직접 블록을 조합하면서, 실시간으로 랜더링 되는 결과화면을 보고, 해당 블록(함수)가 어떻게 작동하는지 이해하도록 하는 것을 목표로 한다.

튜토리얼에서 기본 개념, 2D&3D, 매트릭스 및 카메라 제어, 라이팅 제어 등으로 학습 할 수 있도록 설계했다. 2D&3D 챕터에서는 오브젝트를 그리는 방법을 학습 할 수 있도록 한다. 학습은 단계적으로 이루어지며 2D를 먼저 배운 후 3D를 배우면 좀 더 이해하기 쉽기 때문에 2D 부터 진행한다. 2D 에서는 점, 선, 면을 그리는 방법을 습득한 후 삼각형, 사각형 등 2차원 도형을 그리는 방법을 습득한다. 2D 튜토리얼을 완료하면 직육면체 같은 3D 오브젝트를 만드는 방법을 습득한다.

컴퓨터 그래픽스, 특히 그래픽 파이프라인에서 가장 중요한 개념은 좌표계 변환(Coordinate Transformation)이다. 물체 정점은 물체 하나를 설계할 때의 좌표계, 한 장면에서 여러 문체를 모아놓았을 때의 좌표계, 그 장면을 바라보는 시점에 따른 좌표계 등 여러 가지 좌표계를 거치면서 새로운 좌표 값으로 바뀌어 최종적으로 화면에 그려진다. 좌표계는 크게 물체 별로 설계상의 편의를 위주로 설계된 모델 좌표계와 하나의 장면 안의 모든 물체를 한꺼번에 일률적으로 아우르는 전역 좌표계, 바라보는 화면에 따라 물체의 모습이 달라지는 시점 좌표계로 나뉘어진다. GL 프로그램이 처음 실행될 때는 전역 좌표계와 모델 좌표계가 일치되어 있다. 그러나 오브젝트를 이동, 회전, 크기 조절 등 다양한 변화를 가하면 전역 좌표계와 모델 좌표계가 분리된다. 또 카메라의 이동에 의하여 시점 좌표계도 변할 수 있다. 설계된 물체 또는 그래픽 라이브러리에서 가져온 물체 그대로 장면을 구성하는 데 사용할 수도 있지만, 일반적으로는 장면에 맞도록 변환하는 작업이 필요하다. 이러한 작업을 모델 변환이라고 하며, 일반적으로 물체에 기하 변환을 가하여 물체 모습을 변환하는 작업을 뜻한다. 모델 변환은 뷰 변환과 밀접한 관계를 가지며 뷰 변환은 카메라 조작을 하는 것과 같은 의

미이므로, 카메라 조작 챕터에서 다시 설명하도록 한다. 따라서 이 챕터에서는 오브젝트를 이동, 회전, 크기 조절하는 방법을 학습한다. 상태 변수를 사용하는 GL에서는 항상 현재의 상태 변수, 즉 현 상태 변수 값이 중요하다. 모든 파이프라인 프로세스는 현 상태 변수를 기준으로 가해지기 때문이다. 따라서 상태 변수의 일종인 변환행렬 역시 현재 변환 행렬(CTM : Current Transformation Matrix)값이 중요하다. 다시 말해서 물체 변환은 여러 행렬을 쌓아놓은 스택(Stack) 형태를 취한다. CTM등 행렬의 변환 과정 개념을 가시적으로 보여줌으로써 학습자의 이해를 돕는다. 따라서 내부적인 구현은 스택을 취하고, 사용자 UI에서는 CTM을 변경할 때 마다 3차원 공간에 현재 CTM 좌표를 점을 이용하여 표시하도록 설계한다.



[Fig. 2] CTM Visualization

OpenGL을 사용한 컴퓨터 그래픽스에서는 모델 좌표계의 변화가 현재 변환 행렬에 반영되며, 최종적인 물체는 현재 변환 행렬 기준으로 그려진다. 현재 변환 행렬은 일종의 복합 행렬이다. 복합 행렬은 여러 행렬을 곱한 최종 결과로, 항상 단일 행렬이다. 따라서 최종 결과인 현재 변환 행렬만 가지고는 이전에 어떤 변환 과정을 거쳤는지 짐작할 수 없다. 즉, 모델 좌표계의 변화 과정을 역추적하는 것은 불가능하다. 이에 필요한 것이 행렬 스택(Matrix Stack)이다. 이 스택을 이용하면 좌표계가 변화한 과정을 파악할 수 있다. OpenGL 에서는 glPushMatrix() 와 glPopMatrix() 함수를 사용하여 행렬 스택을 조작하며, 지금 상태의 현재 변환 행렬을 저장하고 싶다면 push함수를 호출하고, 이전에 저장된 변환 행렬을 복원 하고 싶다면 pop함수를 호출한다. 이 개념은 계층 구조 모델링을 하는데 필수적이다. 따라서 앞에서 언급했던 CTM 좌표를 랜더링 공간에 점으로 표시해 주는 방식을 이용하여 사용자의 이해를 돕도록 설계한다.

GL에서 시점 좌표계는 세가지 요소에 의해 정의된다. 카메라 위치, 카메라가 바라보는 점(초점의 위치) 카메라의 기울임(Orientation)이다. 카메라 기울임을 고려해야 하는 이유는 동일한 대상이더라도 카메라를 눕혀서 찍을 수도 있지만 세워서 찍을 수도 있기 때문이다. 따라서 GL에서는 gluLookAt 함수를 사용하여 시점 좌표계를 설정한다. 그러나 gluLookAt 함수는 파라미터를 9개 가지고 있으므로 처음 접해보는 사람 입장에서 이 파라미터들을 제대로 다루기가 어렵다. 따라서 카메라를 조작하는 함수를 좀 더 세분화하여 카메라의 포지션, 초점, 방향을 세부적인 블록으로 나누어 사용자가 좀 더 쉽게 익힐 수 있도록 한다. 투상(Projection) 개념인 Perspective(원근 투상)와 Orthogonal(평행 투상)을 카메라를 이용하여 학습시켰다. 튜토리얼에서 사용자가 perspective 카메라와 orthographic 카메라를 사용하여 투상에 대한 차이점을 직관적으로 확인 할 수 있도록 한다.

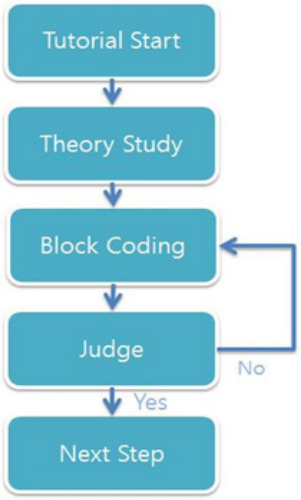
마지막 챕터에서는 빛을 제어하는 방법을 배울 수 있도록 한다. 빛과 조명을 부여하는 것은 물체 정점의 색을 부여하는 작업이다. 즉, 광원과 물체 특성을 감안하여 정점에서의 빛 세기를 계산하는 작업이다. 조명은 컴퓨터 그래픽에서 매우 중요한 위치를 차지한다. 조명에 의해서만 물체 모습이 사실적으로 드러나기 때문이다. OpenGL에서는 이론적인 조명모델을 기반으로 함수 파라미터가 구성되지만, 튜토리얼에서는 빛과 조명의 기본적인 이론과 포지션닝과 밝기, 방향 조절, 종류를 학습하는데 중점을 둔다. 또 앞에서 설명한 바와 같이 복잡한 조명 함수를 블록으로 간소화 시킨다.

3.2.4 평가 시스템

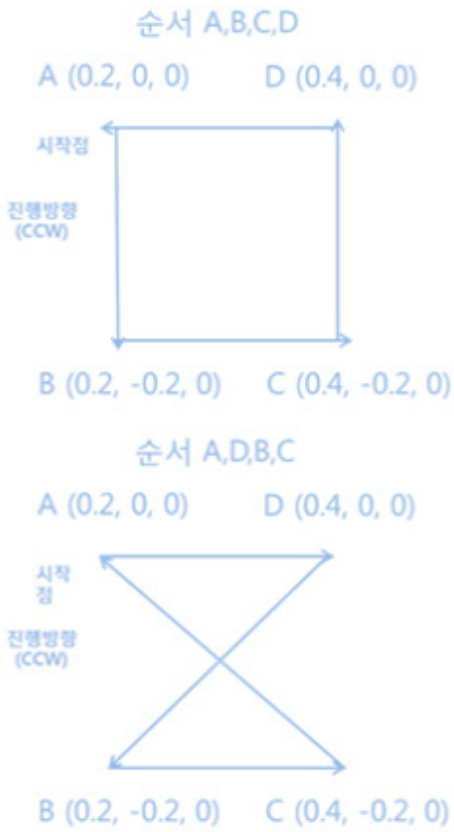
우리의 모델은 교육을 목적으로 하고 있기 때문

에 프로그램을 통하여 사용자가 이론을 이해하고, 튜토리얼을 잘 따라오고 있는지 검증할 수 있는 방법이 필요하다. 따라서 각 튜토리얼 마다 달성해야 할 결과물을 배치하고, 의도대로 블록을 조합했는지 판정을 통하여 검증한다[Fig. 3].

컴퓨터 그래픽스에서 모든 오브젝트는 Vertex(점)의 집합으로 표현 할 수 있다. 이러한 점들은 각각 X, Y ,Z 좌표를 갖고 있으며 찍는 순서에 따라 오브젝트의 앞, 뒤를 나타낸다. 이를 이용하여 오브젝트 그리는 방법을 학습시키고 판정에 활용한다. 예를 들면, 삼각형은 세 점 A, B, C의 집합으로 표현 할 수 있다. 이에 따른 세 점의 조합 순서는 2^3의 가짓수로 나타낼 수 있다. 만약 여기서 삼각형의 첫 시작점 A를 주고 오브젝트를 그리는 순서가 CCW 방식임을 알고 있다면 결국 올바른 답은 반 시계 방향으로 A, B, C 순으로 점을 나타내는 것임을 판단 할 수 있다. 삼각형의 경우 순서가 잘못 되어도 크게 문제가 되는 경우는 없지만 사각형의 경우 그려지는 모양이 크게 달라진다[Fig. 4]. 따라서 Vertex를 찍는 순서를 체크하여 의도대로 파라미터 값을 넣었는지 판정한다. 빛이나 카메라 판정의 경우 블록 파라미터에 우리가 의도한 범위 내의 값을 넣었는지 판단한다.



[Fig. 3] Evaluation Process

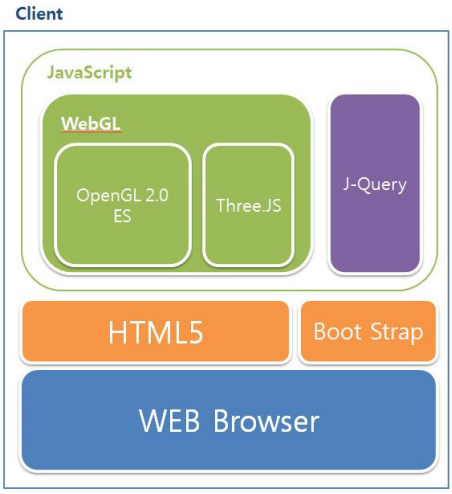


[Fig. 4] Different Results by Drawing Order

3.3 시스템 구조

3.3.1 Front-End 구조

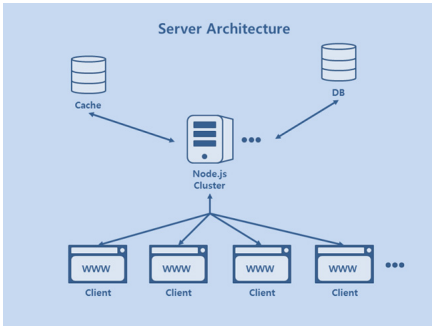
Front-End는 웹 브라우저에서 돌아가기 때문에 HTML5를 기반으로 구축한다. 레이아웃 구성은 반응형 웹을 지원하는 부트스트랩을 사용하며 그래픽스 구현에는 WebGL을 사용한다. WebGL은 간단히 말하면 OpenGL의 웹 버전으로, OpenGL의 기능 축소판인 OpenGL ES 2.0을 기반으로 웹 브라우저에서 컴퓨터 그래픽스를 구현하는데 주로 사용된다[9]. 또 WebGL을 쉽게 구현할 수 있도록 도와주는 라이브러리인 Three.js를 사용한다[Fig. 5].



[Fig. 5] Front-End System Architecture

3.3.2 Server-Side 구조

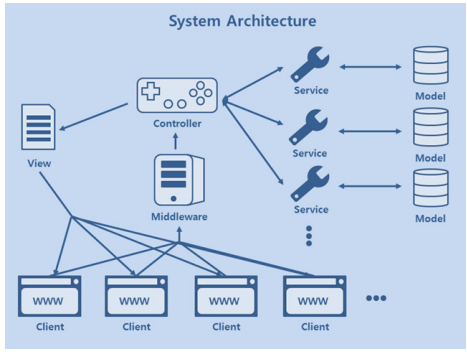
서버의 구성으로는 Node.js[10]와 Express Web Framework를 이용하여 구성하며 사용자의 데이터나 Application 구현에 있어 필요한 데이터를 저장하기 위해 RDB(Relational Database)인 MySQL을 사용한다. Node.js가 Single-Thread 로 운영되기 때문에 System utilization 을 높이기 위하여 Node.js built-in cluster 모듈을 이용하여 Application을 clustering하여 운영한다[Fig. 6].



[Fig. 6] Node.js Server Architecture

Node.js 는 Single-Thread 로 작동하기 때문에 클러스터로 구성해도 서로 독립적인 어플리케이션으로 운영된다. 때문에 섹션(session) 정보나 공유가 필요한 정보를 관리하기 위하여 외부 저장소를 이용한다. 이 외부저장소를 이용하여 각각의 클러스터들이 정보를 주고받는다. 하드 디스크를 이용하는 데이터베이스의 경우에는 이러한 공유 정보 관리에 있어서 성능 문제가 있어 제외하며 대신에 In-Memory Cache를 사용한다.

시스템 소프트웨어 구성으로는 기본적으로 MVC(Model-View-Controller) 패턴을 이용하여 구성한다. 추가적으로 MVC 패턴의 컨트롤러 부분의 방대해짐과 코드 중첩 문제를 해소하기 위하여 Service Layer를 추가하여 해결하며 사용자 인증 처리, 로그, 에러처리는 미들웨어(Middleware)를 이용한다. 클라이언트 요청 시 Node.js Route Dispatcher가 적합한 미들웨어를 선택하여 선행적으로 처리되어야 할 부분들(로깅/인증)을 처리한 뒤 컨트롤러를 호출한다. 컨트롤러는 필요한 서비스들을 호출하여 얻어낸 정보를 View에 업데이트 후 응답함으로써 요청 처리를 종료한다[Fig. 7].

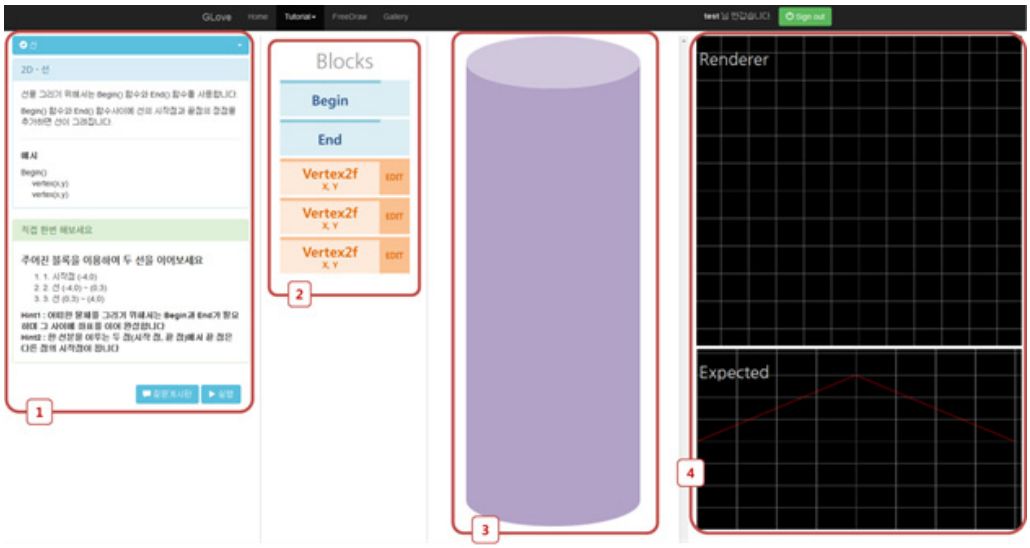


[Fig. 7] MVC(Model-View-Controller) Design Pattern

4. 결 과

4.1 튜토리얼 페이지 UI

튜토리얼 페이지는 총 4개의 섹션으로 구성되어 있다[Fig. 8]. 첫 번째 섹션에서는 해당 챕터에서



[Fig. 8] Tutorial User-Interface

배울 수 있는 그래픽스 이론에 대한 설명과 어떤 블록을 어떻게 조합해야 되는지에 대한 대략적인 설명을 확인할 수 있다. 사용자는 블록들을 조합하여 4번째 영역의 Expected에 나타난 화면과 똑같이 Renderer 화면을 만들어 주어야 한다.

가운데 영역에서는 해당 챕터에서 사용할 수 있는 블록들이 정렬되어 있으며 2번 영역의 블록을 3번 영역으로 드래그 해서 블록들을 조합할 수 있다. 블록의 파라미터를 넣을 땐 EDIT 버튼을 눌러 나타나는 모달(Modal) 창에 값을 입력하면 넣을 수 있다. 3번째 영역에 블록을 넣으면 4번째 영역의 Viewport 영역에서 넣은 블록의 종류에 따라 실시간으로 랜더링 되는 모습을 확인할 수 있다.

Viewport 화면과 Expected 화면을 일치시킨 후 1번 영역에 있는 실행 버튼을 클릭하면 판정 알고리즘이 작동한다. 올바른 블록 조합순서와 파라미터를 넣었을 경우 클리어 화면과 함께 다음 챕터로 넘어갈 수 있다. 반면 잘못 넣었을 경우 어디에서 틀렸는지 사용자가 체크할 수 있도록 안내 메시지를 보여준다. 또 부가적으로 각 챕터마다 질문 게시판을 제공하여 사용자들끼리 정보를 공유할 수 있도록 하였다.

4.2 튜토리얼 구성

[Table 1] Block Type of Tutorial

idx	ID	Parameter
1	begin	NULL
2	end	NULL
3	vertex2	x, y
4	vertex3	x, y, z
5	drawBox	size, x, y, z
6	drawSphere	R, Lo, La
7	pushMatrx	NULL
8	popMatrix	NULL
9	translate	x, y, z
10	rotate	theta, x, y, z
11	scale	x, y, z
12	perspective	fov, near, far
13	orthographic	left, right, top, bottom, near, far
14	lookAt	x ,y ,z
15	directionalLight	hex
16	spotLight	hex, intensity, distance, angle
17	camreaPosition	x, y, z
18	lightPosition	x, y, z
19	identityMatrix	NULL
20	lightDirection	x, y, z

튜토리얼 구현은 총 16 단계로 이루어져 있으며 튜토리얼에서 사용할 수 있는 블록은 총 20개이다[Table 1]. 각 블록들은 각기 다른 하나의 함수에 맵핑(Mapping)되어 렌더링 기능을 수행한다. 변환(이동, 회전, 크기조절)블록들은 순서에 민감하여 순서를 변경하는 것만으로도 다른 결과를 보여준다.

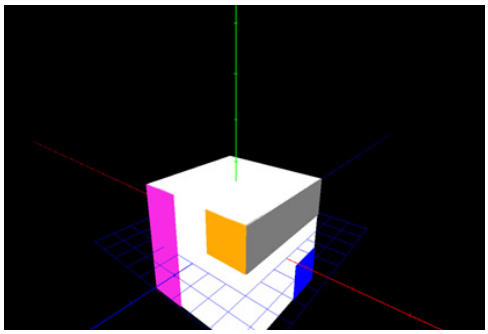
각 챕터 마다 필요한 블록들이 다르다[Table 2]. 이에 따라서 위의 표와 같이 챕터별 블록을 정리하여 데이터베이스에 저장하고 있으며, 챕터가 로딩 될 때 챕터 정보와 블록 정보를 맵핑하여 클라이언트에 로드 된다. 각 챕터의 블록을 정확히 기능을 구현하는데 필요한 블록의 종류와 개수만 불러와 모든 블록을 사용하여 기능을 구현하도록 유도하였다. 각각의 블록의 파라미터들은 허용되는 타입이 정해져 있으며, 해당 타입을 벗어나는 값은 넣지 못하도록 정규식을 사용하였다.

[Table 2] Block List

Chapter	Blocks
Vertex	vertex 3
Line	begin 1, end 1 ,vertex 3
Triangle	begin 1, end 1 ,vertex 3
Rectangle	begin 1, end 1 ,vertex 4
Low API Box	begin 2, end 2, vertex 8
High API Box	drawBox 1
High API Sphere	drawSphere 1
Translate	drawBox 3, translate 3, idenity Matrix 1
Rotate	drawBox 1, translate 1, rotate 1, idenity Matrix 1
Scale	drawBox 1, scale 1, idenity Matrix 1
Push, Pop	psuh 2, pop 2, drawBox 3, translate 2
Perspective	drawBox 1, perspective 1
Orthographic	drawBox 1, perspective 1, orthogonal 1
Position, LookAt	drawBox 3 , camera position 1, lookAt 1
Directional Light	directional light 1, lightDirecton 1, light position1
Spot Light	spot light 1, lightDirecton 1, light position 1

4.3 오브젝트 모델링

튜토리얼을 모두 클리어하면 자유롭게 블록을 이용하여 사용자가 원하는 오브젝트를 모델링 할 수 있도록 하였다. 사용하는 블록과 방식은 튜토리얼과 같으며, 조합된 블록 순서와 파라미터 값을 OpenGL이나 Processing에서 그대로 문자로 코딩할 경우 같은 결과를 얻을 수 있다. 따라서 블록을 이용하여 이론을 학습하는 것만이 끝이 아닌, OpenGL 같은 프로그래밍 라이브러리를 사용하는데 도움이 될 수 있도록 연계하였다[Fig. 9].



[Fig. 9] Modeling Using Block

5. 결 론

우리는 언제 어디서나 웹을 통해 접근하여 블록 조합을 이용해 쉽게 컴퓨터 그래픽스를 학습할 수 있는 소프트웨어 모델을 제시 했다. 이론 중심으로 학습함으로써 프로그래밍 언어에 종속되지 않고, 환경에 맞추어 그래픽스 프로그래밍 할 수 있다. 더욱이 우리 모델은 블록을 조합할 때 마다 실시간으로 랜더링 결과를 보여줌으로써 사용자들이 특정 블록(함수)가 어떤 기능을 하는지 직관적으로 이해 할 수 있다. 또 컴파일 에러를 신경 쓸 필요가 없으므로 학습에 집중 할 수 있다. 그러나 그래픽스 이론에 중점을 뒀기 때문에 언어적 특성이 강한 조건, 반복문은 제외시켰다. 따라서 조건, 반복문을 사용하는 애니메이션 기능도 함께 제외되었

지만 우리의 모델을 이용해 학습한 이론과 언어를 조금만 응용하면 충분히 구현할 수 있을 것이다.

개발 기간 동안 내부 알파테스트와 프로그래밍 언어를 배운 사용자 20여명을 대상으로 베타테스트를 실시했다. 컴퓨터 그래픽스를 조금이라도 접해 본 사용자들은 블록을 통해 쉽게 개념적 지식을 이해하고, 자신의 개념이 맞는지 실시간 랜더링을 통해 확인할 수 있었으며, 처음 접하는 사용자들도 튜토리얼을 마친 후 자신이 원하는 오브젝트를 만들 수 있는 수준까지 학습할 수 있었다. 블록 기반 시스템이 사용자로 하여금 흥미를 유발하였다는 평가와 즉각적 반응, 단계적 결과 도출로 인해 몰입도가 향상되었다는 평가가 주류를 이루었다. 하지만 UI적인 측면에서 직관성이 부족하다는 지적이 있어 향후에 보완할 계획이다. 아울러, 이론 내용, 직관적인 사용자 인터페이스 개선, 정량적 테스트 등의 문제는 향후 연구 과제로 포함시켜 발전시켜야 할 필요성이 있다. 앞으로 우리가 만든 교육용 모델을 통해 컴퓨터 그래픽스 분야에 대한 관심이 증가하고, 쉽게 그래픽스 이론을 배움으로써 컴퓨터 그래픽스에 대한 접근성이 높아지는 효과를 기대한다.

REFERENCES

[1] Sang Kwon Goo, “Computer Graphics, Intuition and Idea”, Journal of Korean Society of Media and Arts, Vol. 11, No. 1, pp77-88, 2013.

[2] Mitchel Resnick, “Scratch: programming for all”, Communications of the ACM, Vol. 52, No. 11, pp60-67, 2009.

[3] Jeong-Beom Song, Soeng-Hwan Cho, Tae-Wuk Lee, “The Effect of Learning Scratch Programming on Students' Motivation and Problem Solving Ability”, Korea Association of information education, Vol. 12, No. 3, pp323-332, 2008.

[4] Kyeong Mi Ahn, Won-Sung Sohn, Woon-Chul Choy, “The Effect of Scratch Programming Education on Learning-Flow and Programming Ability for Elementary Students”, Korea Association of information education, Vol. 15, No. 1, pp1-10, 2011.

[5] Seong-Hwan Cho, Jeong-Beom Song, Seong-Sik Kim, Kyung-Hwa Lee, “The Effect of CPS-based Scratch EPL on Problem Solving Ability and Programming Attitude”, Korea Association of information education, Vol. 12, No. 1, pp77-88, 2008.

[6] John Maloney, “Programming by choice: urban youth learning programming with scratch”, ACM SIGCSE Bulletin - SIGCSE 08, Vol. 40, No. 1, pp367-371, 2008.

[7] B Chen, HH Cheng, “Interpretive OpenGL for computer graphics”, Computers & Graphics, Vol. 29, No. 3, pp331-339, 2005.

[8] Edward Angel, “Teaching a three-dimensional computer graphics class using openGL”, ACM SIGGRAPH Computer Graphics, Vol. 31, No. 3, pp54-55, 1997.

[9] John Congote, “Interactive visualization of volumetric data with WebGL in real-time”, Proceedings of the 16th International Conference on 3D Web Technolog, pp137-146, 2011.

[10] Stefan Tilkov, Steve Vinoski “Node. js: Using JavaScript to build high-performance network programs”, Computers & Graphics, Vol. 14, No. 6, pp80-83, 2010.

편 해 걸(Hae-Gul Pyun)

2010년 숭실대학교 글로벌미디어 학사

관심분야 : 컴퓨터 그래픽스, 웹 프로그래밍

박 진 호(Jinho Park)

1999년 KAIST 수학과 졸업(이학사)
2001년 KAIST 응용수학과 졸업(이학석사)
2007년 KAIST 전산학과 졸업(공학박사)
2007년-2008년 KAIST 문화기술 대학원 박사 후 연구원
2008년-2009년 UCLA 전산학과 박사 후 연구원
2009년-2013년 남서울대학교 멀티미디어학과 조교수
2013년-현재 숭실대학교 글로벌미디어학부 조교수

관심분야 : 컴퓨터 그래픽스, 물리 기반 애니메이션