**OpenGL 초기화**

OpenGL의 초기버전에서 사용한 함수 세트는 현대 OpenGL의 함수 세트와는 매우 다르다.

OpenGL은 3개의 프로파일을 지원한다.(코어, 호환성, ES)

파라미터 설정을 위해 SDL\_GL\_SetAttribute(설정할 속성,해당 속성의 값); 함수를 호출한다.

Context는 OpenGL이 인식하는 모든 상태나 오브젝트를 포함하는 OpenGL의 세계이다. 색상깊이, 로드된 이미지나 모델, 그리고 여러 다양한 OpenGL 오브젝트를 포함한다.

윈도우를 생성하고 제거하는 것처럼 OpenGL 콘텍스트를 소멸자에서 제거해야 한다.

**삼각형 기초**

현대의 그래픽 하드웨어에서는 폴리곤 렌더링이 매우 효율적이어서 2D이든 3D이든 최근의 모든 게임에서는 궁극적으로 폴리곤을 사용한다.

다른 3D 그래픽 테크닉과 비교해보면 폴리곤은 런타임 시 많은 계산이 필요하지 않다.

크기를 가변적으로 조절할 수 있다.

대부분의 3D오브젝트를 폴리곤으로 표현할 수 있다.

그 중 삼각형은 3개의 버텍스만 필요로 하고 그렇기 때문에 한 평면안에만 놓일 수 있다.

삼각형은 쉽게 테셀레이션이 가능하다(복잡한 3D물체를 여러 개의 삼각형으로 쉽게 나눌 수 있다.)(다른 폴리곤(ex. 사각형)이 한 평면상에 놓일 수 있다면 다른 폴리곤도 사용 가능)

정규화된 장치 좌표

OpenGL에서는 정규화된 장치 좌표를 사용한다. 윈도우의 중심이 장치좌표의 중심

버텍스 버퍼와 인덱스 버퍼

여러 갱의 삼각형으로 구성된 3D 모델이 있다고 하면 메모리상에 저장해야 하는데 가장 간단한 방법은 인접한 배열이나 버퍼 형태로 직접 저장하는 것이다.

하지만 버텍스의 배열은 일부 중복 데이터를 갖고 있다. 해결 방법으로 버텍스 버퍼에는 3D 기하에서 사용한 고유한 좌표만을 담고 버퍼에 인덱스를 붙인다. 인덱스 버퍼는 인덱스 3개로 구성된 개별 삼각형 정보를 배열 형태로 저장한다.

**셰이더**

현대의 그래픽스 파이프라인은 단순히 버텍스/ 인덱스 버퍼만 제공받아서 삼각형을 그리는 것이 아니라 삼각형은 고정된 색상으로 그릴 것인지, 텍스처에서 얻은 색상을 버텍스에 사용할 것인지, 그려야하는 모든 픽셀에 광원 계산을 할 것인지 등 어떻게 버텍스를 그려야 할지 지정하는 작업이 필요하다.

셰이더 프로그램은 그래픽 하드웨어 상에서 특정 태스크를 수행할 때 실행되는 작은 프로그램으로 자신만의 메인 기능을 가진 별도의 프로그램이다.

버텍스 셰이더

버텍스 셰이더 프로그램은 그려질 모든 삼각형의 모든 버텍스에 대해 한 번 씩 실행된다.

버텍스 셰이더는 입력으로 버텍스 속성 데이터를 받아서 적절하게 수정한다.

프래그먼트 셰이더

삼각형의 버텍스가 버텍스 셰이더를 거친 후에 OpenGL은 삼각형에 해당하는 픽셀이 어떤 색상을 가지는지를 결정해야 한다. 삼각형을 픽셀로 변환하는 과정을 래스터 변환이라 한다.

프래그먼트 셰이더의 역할을 각 픽셀의 색상을 결정하는 것이다. 따라서 모든 픽셀마다 한 번씩 실행된다. 이 색상은 텍스처나 색상, 재질 같은 표면 속성을 고려해서 결정된다.조명이 존재한다면 광원계산도 고려해야한다.

셰이더 로딩

고수준 레벨에서는 다음과 같은 단계를 수행해야 한다.

1. 버텍스 셰이더를 로드하고 컴파일 한다.
2. 프래그먼트 셰이더를 로드하고 컴파일한다.
3. 2개의 셰이더를 ‘셰이더 프로그램’에 서로 연결시킨다.

**변환의 기초**

한 유형의 오브젝트에 대해 여러 번 사용하는 경우(위치나 회전값이 다른 상태로) 각 오브젝트에 버텍스 버퍼를 하나씩 대응시키는 것보다 하나의 버텍스 버퍼를 재사용하는 것이 효율적이다.(2D나 3D 모두)

오브젝트 공간

3D 모델링 프로그램을 사용해서 3D 오브젝트를 만들 때 일반적으로는 정규화된 장치 좌표로 버텍스 위치를 나타내지 않는다. 대신 위치는 오브젝트 그 자체의 임의의 원점에 상대적이다.(오브젝트의 중심) 이 오브젝트 그 자체에 대한 상대적인 좌표 공간을 오브젝트 공간 또는 모델공간 이라고 한다.

게임을 실행하면 각각의 고유모델은 자신의 버텍스 배열 개체(VAO)에 로드되고 장면을 그릴 때 각 오브젝트의 버텍스는 버텍스 셰이더로 전달된다. 하지만 버텍스 위치를 수정없이 직접 프래그먼트 셰이더에 넘긴다면 문제가 된다. 이 버텍스 위치는 정규화된 장치 좌표로서 기대되지만 직접 넘긴 버텍스 모델 좌표는 NDC가 아니라 오브젝트의 상대적인 좌표이다. 그대로 넘기면 안된다.

세계 공간

각 인스턴스를 그릴 때는 각 오브젝트에 대해 같은 버텍스 배열 개체를 사용한다.

그러나 이제 각 인스턴스는 오브젝트 공간 좌표를 세계공간으로 변환하는 방법을 지정하는 추가적인 정보가 더 필요하다. 이 여분의 데이터는 인스턴스를 그릴 때 버텍스 셰이더로 보낼 수 있으며, 버텍스 셰이더가 필요에 따라 버텍스 좌표를 보정하는데 쓰인다.

세계 공간으로 변환

좌표공간을 변환할 때는 두 좌표 공간 사이에 기저 벡터가 같은지 또는 그렇지 않은지를 알아야 한다.(+y가 두 좌표공간에서 동일한 방향인지 등)

변환할 때 한가지 방법은 정확한 버텍스 위치를 계산하기 위해 대수 방정식을 사용하는 것이다.

이동

이동은 점을 변환시키거나 오프셋값으로 이동시킨다.

점(x,y)가 주어졌을 때 다음 방정식을 사용하면 오프셋 (a,b)만큼 점을 이동시킬 수 있다.

X’ = x+a, Y’ = y +b

스케일

스케일을 삼각형에 적용하면 삼각형의 크기는 커지거나 작아진다.

균등 스케일에서는 버텍스의 각 요소에 같은 스케일 팩터 s를 사용해서 스케일한다.

X’ = x \* s, Y’ = y \* s

비균등 스케일에서는 각 요소에 대해 별도의 스케일 팩터(sx, sy)를 곱한다.

회전

사인과 코사인을 사용하면 다음과 같이 임의의 점 (x,y)를 각도 θ 만큼 회전시키는 공식은 다음과 같다.

X’ = xcos θ - ysin θ, Y’ = xsin θ + ycos θ

단위 원에서 처럼 각도 θ는 반시계방향 회전을 나타낸다. 회전은 원점기준이다.

변환을 결합

앞의 수식들은 각 변환을 독립적으로 적용하지만 동일한 버텍스에 대해서는 여러변환을 동시에 적용하는 것이 일반적이다.

변환 순서가 중요하므로 일관성 있는 순서를 갖는 것이 중요하다. 오브젝트 공간에서 세계공간으로의 변환에서는 항상 스케일, 회전, 이동순으로 변환을 적용하자.

방정식 통합에 따른 문제

오브젝트 공간에서 임의의 버텍스를 얻어서 각 요소에 방적식을 적용하면 세계공간으로 변환된 임의의 스케일, 회전, 위치를 가진 버텍스를 얻을 수 있었다. 하지만 변환만 할 뿐 세계 공간의 좌표는 장치 좌표에 대해 정규화 되어 있지 않으므로 여전히 버텍스 셰이더에서는 많은 변환을 적용해야 한다.

**행렬과 변환**

행렬은 m행과 n열이 있는 값의 격자다. 컴퓨터 그래픽스 분야에서는 행렬을 사용해서 변환을 표현한다.

행렬 곱셈

= (왼쪽행렬의 열의 수는 오른쪽 행렬의 행의 수와 같아야 한다.)

행렬 곱셈은 결합은 가능하지만 교환은 가능하지 않다. (AB != BA), A(BC) = (AB)C

행렬을 사용한 점의 이동

변환측면에서 보면 행렬은 임의의 점으로 표현하는 것이 가능하다. (행 또는 열로 표현 가능) 일관된 하나의 방식을 사용하는 것이 중요하다.

변환행렬 T가 있다고 할 때 점p가 행일 때 와 열일 때 곱한 결과가 서로 다르다.

컴퓨터 그래픽스에서는 리소스나 그래픽 API에 따라 행 벡터 혹은 열 벡터를 사용한다.

트랜스포즈를 각 변환행렬에 적용하면 행 벡터를 열 벡터로, 혹은 그 반대로 전환하는 것이 가능하다.

[ =

항등행렬은 대문자 I로 표시되는 특별한 유형의 행렬이다. 같은 수의 행과 열을 가지고 대각선은 1을, 대각선을 제외하고는 0의 값을 갖는다. 항등행렬은 임의의 행렬과 곱하면 행렬은 변경되지 않는다.

세계 공간으로 변환, 재검토

스케일 행렬

스케일 변환을 적용하기 위해 2X2 스케일 행렬을 사용한다.

S(sx,sy) =

회전 행렬

2D 회전 행렬은 θ로 z축에 대한 회전을 나타낸다. R(θ) =

이동 행렬

2X2 행렬은 2D 스케일 및 회전 행렬의 표현이 가능하지만 2X2크기로는 일반적인 2D 이동 행렬을 표현할 방법이 없다. 유일한 방법은 3X3 행렬로 표현하는 것이다.

T(a,b) =

그러나 1X2행렬은 충분한 열을 가지고 있지 않아서 3X3과 곱할 수 없기 때문에 행 벡터에 추가 열을 더해서 1X3 행 벡터로 만드는 것이 유일한 방법이다.

동차좌표는 n차원 공간을 나타내기 위해 n+1 요소를 사용하기 때문에 2D는 3개요소를 사용한다. 추가된 세번째 요소는 3D공간을 나타내기 위함이 아니기 때문에 w요소라 한다. (x,y,w)

변환 결합

스케일 및 회전 변환을 동차 좌표로 동작하는 3X3 행렬로 표현해야 한다.

오브젝트 공간에서 세계 공간으로 변환하는 이 결합 행렬은 세계 변환 행렬이다.

스케일, 회전, 이동 순으로 곱한다. S(sx,sy) \* R(θ) \* T(a,b)

세계 변환을 액터에 추가