사례 발표

모바일 플랫폼상 3D 라이브러리의 개발

박곤호[1], 양종열[2], 한우진[3], 안민수[4], 이승용[5]

목 차

1. 서 론

모바일 폰 기술이 급격히 발달함에 따라 사용자의 요구가 다양해지면서, 단순히 통화 기능뿐만 아니라 게임 등의 화려한 그래픽 컨텐츠를 선호하게 되었다. 따라서 모바일 그래픽 기술 개발의 필요성이 대두되었다. 모바일 플랫폼에서 그래픽 기술이 많이 적용되는 대표적 어플리케이션으로 모바일 게임이 있지만, 현재 하드웨어의 한계로 인하여 영상 기반의 그래픽으로만 제작되고 있다. 그러나 시장의 추세와 기술 성장으로 모바일 폰에서도 삼차원 그래픽이 대중화될 전망이다. 모바일 플랫폼의 삼차원 기술은 게임뿐만 아니라 애니메이션이나 아바타 등 여러 어플리케이션에서 유용하게 쓰일 것이다.

본 논문에서는 모바일 환경에서 삼차원 어플리케이션을 만들기 위한 기반 기술로써 삼차원 그래픽 라이브러리를 구현한다. 물론 현재 구현된 삼차원 그래픽 라이브러리가 없는 것은 아니지만, 그 기능이 축소되어 있기 때문에 응용하기가 힘들다. 본 논문에서 구현한 라이브러리는 Java 3D 클래스 구조를 응용했으며, 기존의 모바일 삼차원 그래픽 라이브러리에서 제공하지 않았던 광원(light source), 그로 쉐이딩(Gouraud shading), 텍스처 매핑(texture mapping) 등의 고급 렌더링 기능을 제공한다. 또한 객체 지향을 고려하여 설계했기 때문에 프로그램 개발이 쉽다. 본 논문의 구성은 2절에서 삼차원 위치 변환, Java 3D 등의 배경 지식에 대하여 설명하고, 3절에서는 구성 요소를 설명한다. 4절에서는 성능 평가를 보여줄 것이다. 5절에서는 결론과 향후 연구 방향을 제시할 것이다.

1) (주)일렉트릭아일랜드 사장
2) (주)일렉트릭아일랜드 개발이사
3) 포항공과대학교 석사과정
4) 포항공과대학교 박사과정
5) 포항공과대학교 조교수

2. 배경 지식

2.1 삼차원 위치 변환 (3D Transformation)

삼차원 위치 변환은 그래픽 응용뿐만 아니라, 모든 삼차원 그래픽 엔진의 주요 요소로서 이동, 크기 변화, 회전으로 구성되어 있다. 삼차원 컴퓨터 그래픽스 시스템에서의 삼차원 공간의 위치를 (x, y, z)대신 (x, y, z, w)로 표현하는 좌표를 동

차좌표라 한다(수식 1 참조). 이렇게 동차 좌표를 사용하는 가장 큰 이유는 삼차원 위치변환을 행렬과 벡터의 곱셈으로 통일하려는 것이다. 본 논문의 모든 삼차원 공간좌표는 동차좌표로 표현하였다. 삼차원 공간의 점(x, y, z)을 (d_x, d_y, d_z)만큼 이동한 후의 좌표와 이동 행렬은 (수식 2)와 같다. 삼차원 공간의 점(x, y, z)을 원점을 중심으로 (S_x, S_y, S_z)만큼 크기 변환시킨 좌표와 행렬은 (수식 3)과 같다. 삼차원 공간의 정점을 각 축에 대하여 회전시킬 때 사용되는 행렬은 수식은 (수식 4)와 같다. R_x는 x축 중심, R_y는 y축 중심, R_z는 z축 중심으로 회전하는 변환 행렬이다.

$$(x, y, z, w) = (\alpha x, \alpha y, \alpha z, \alpha w) \quad \text{(수식 1)}$$

$$T(d_x, d_y, d_z)[x \quad y \quad z \quad 1]^T = [x + d_x \quad y + d_y \quad z + d_z \quad 1]^T$$

$$T(d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(수식 2)}$$

$$T(s_x, s_y, s_z)[x \quad y \quad z \quad 1]^T = [xs_x \quad ys_y \quad zs_z \quad 1]^T$$

$$T(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(수식 3)}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(수식 4)}$$

2.2 Java 3D

본 논문에서 구현하는 삼차원 그래픽 엔진은 Java 3D를 모델로 구현하였다. Java 3D는 JDK 1.2버전부터 Sun사에서 선택적 패키지로 제공하는 삼차원 라이브러리이다. Java 3D를 이용함으로써 얻게 되는 장점은 객체지향 설계를 통한 개발 시간의 단축, 애플릿을 이용한 네트워크 환경의 이용, 그리고 플랫폼에 독립적인 특성으로 인한 성능 확장의 용이 등이다.

3. 구현한 삼차원 라이브러리 구성 요소

3.1 삼차원 물체 표현 방식

본 논문의 삼차원 라이브러리에서는 삼차원 물체 표현을 위해 다각형 메쉬 중에서도 삼각 메쉬(triangular mesh)를 이용하여 구현했다. 다각형 메쉬는 다각형을 연속적으로 연결하여 삼차원 물체를 표현하는 방식으로서, 그 표현 방법이 간단하고 어떤 곡면이든 근사할 수 있기 때문에 널리 사용되고 있다.

〈표 1〉 삼차원 물체 표현을 위해 구현한 클래스 및 메소드

구현 클래스	클래스 내용
Shape3D	· 물체에 대한 텍스처 정보를 저장하는 Appearance 객체와 물체의 기하 정보를 저장하는 Geometry 객체를 멤버로 가진다. · 이와 관련된 함수로는 setGeometry, setAppearance, getGeometry, getAppearance 메소드 등이 있다.
GeometryArray	· 꼭지점 위치, 색, 법선 벡터, 텍스처 좌표 정보와 면 정보를 저장 · 꼭지점 위치는 Coordinate, 색은 Col, 법선 벡터는 Normal, 텍스처 좌표는 Texture 멤버이며, 각각에 대한 포인터 역할을 하는 정수형 멤버들이다. · 본 삼차원 라이브러리에는 삼각 메쉬만 지원하므로, GeometryArray 클래스의 하위 클래스는 삼각형에 대한 정보를 저장하는 TriangleArray 클래스만 구현하였다. · 이와 관련된 함수로는 setCoordinate, setColor, setNormal, setTextureCoodinate, getCoordinate, getColor, getNormal, getTextureCoodinate 메소드 등이 있다

3.2 뷰잉 파이프라인

삼차원 물체는 뷰잉 파이프라인(viewing pipeline) 처리를 거쳐서 화면에 그릴 수 있도록 이차원 상의 위치를 변환해야 한다. 즉, 삼차원 물체가 놓여 있는 모델링 좌표계는 뷰잉 파이프라인을 통해 화면 좌표계로 전환한다. 뷰잉 파이프라인 작업을 도식화하면 (그림 1)과 같다.

〈표 2〉 뷰잉 파이프라인을 위한 클래스와 메소드

구현 클래스	클래스 내용
ViewingPipeline	· 삼차원 물체를 화면에 보여주는 작업을 위한 정적 메소드들로 구성된다. · render 메소드 - 뷰잉 파이프라인 작업의 시작점. >Sharp3D 객체를 인자로 받아 Geometry 객체를 추출하고, 투영 변환 행렬, 뷰잉 변환 행렬, 모델 변환 행렬을 곱하여 각 꼭지점에 곱해져야 할 변환 행렬을 구한다. 렌더링 모드를 나타내는 변수에 따라 화면에 drawWireFrame, drawFlat, drawGourand 메소드중 하나를 호출하여 앞에서 추출된 Geometry 객체와 행렬 곱셈을 통하여 구해진 변환행렬을 인자로 넘긴다. · initialize 메소드 - render 메소드가 실행되기 전에 호출되어 각 변환행렬 및 깊이 버퍼를 초기화한다. · transform 메소드 - 물체를 구성하는 각 꼭지점마다 render 메소드에서 구한 변환행렬을 곱하여 각 꼭지점의 정규화된 Device 좌표계를 구한다. · setLookAt 메소드 - 카메라를 설정한다. · setFrustum 메소드 - 뷰 볼륨의 모양을 설정한다

3.3 조명 계산

삼차원 물체를 사실적으로 표현하기 위해서는 그 물체가 발산하거나 반사하는 빛을 고려해야 한다. 물체의 재질 및 빛의 위치, 세기 등에 따라 우리의 눈이 인지하는 정보는 달라진다. 컴퓨터 그래픽스에서는 이러한 것을 고려하여 조명 모델을 제시하였다. 이 절에서는 본 논문에서 구현한 조명 모델에 대해서 살펴보도록 하겠다.

〈표 3〉 조명 계산을 위한 클래스와 메소드

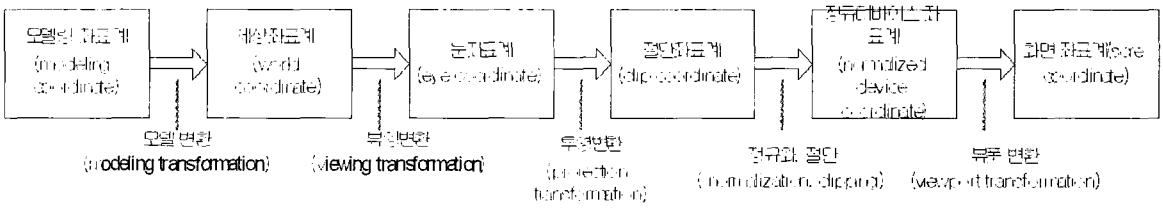
구현 클래스	클래스 내용
Shading	· illuminate 메소드 - Shading 클래스의 멤버 함수 >난 반사광 모델을 이용하여 빛의 강도를 구한다. >빛의 강도는 광원의 위치를 가리키는 좌표에서 꼭지점의 위치를 가리키는 좌표를 빼서 구한 벡터와 꼭지점의 법선 벡터를 내적하여 빛의 강도를 구한다.
ViewingPipeline	· setPLight, getPLight 메소드 - ViewingPipeline 클래스의 멤버 함수 >광원의 위치를 지정하거나 얻는다. >본 논문의 삼차원 라이브러리는 광원이 하나만 존재하도록 구현했다.

3.4 쉐이딩

사실적인 영상을 만들기 위해서는 물체 고유의 색과 조명의 위치 및 방향을 고려하여 각 다각형 안을 적절한 색으로 채워야 한다. 이렇게 다각형의 색상을 결정하는 것을 쉐이딩이라고 한다. 본 논문에서는 삼차원 라이브러리에서는 널리 사용하고 있는 플랫 쉐이딩과 그로 쉐이딩을 구현하였다.

〈표 4〉 쉐이딩을 위한 클래스와 메소드

구현 클래스	클래스 내용
Shading	· fillTriangleFlat 메소드 >Point4f 배열 객체, Color 객체, Graphics 객체를 인자로 받아 삼각형을 하나의 색으로 채운다. · fillTriangleGouraud 메소드 >보간을 수행하여 결정된 색으로 삼각형을 채운다



(그림 1) 뷰잉 파이프라인 처리 과정

3.5 은면 제거

화면에서 보이지 않은 다각형들은 제거하는 것을 은면 제거라고 한다. 은면 제거의 방법에는 Z-버퍼 알고리즘과 후면 선별이 있고, 본 논문은 이 두 가지 알고리즘을 모두 이용하여 은면 제거를 구현하였다.

〈표 5〉 은면 제거를 위한 클래스와 메소드

구현 클래스	클래스 내용
ViewingPipeline	· backFaceCull 메소드 ▷후면 제거 연산을 위한 메소드 ▷법선 벡터와 카메라로부터 면을 향하는 벡터를 이용한다. ▷두 각이 90도를 넘으면 그 면은 뷰잉 파이프라인에서 제외된다.

3.6 텍스처 매핑(Texture Mapping)

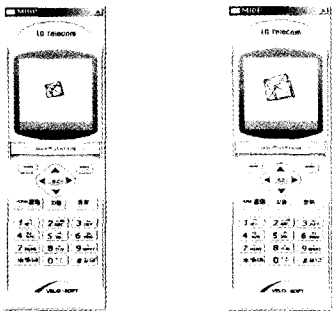
물체의 사실성을 높이기 위해서 본 논문의 삼차원 라이브러리는 텍스처 매핑 방법을 적용했다. 이 방법은 다각형 메쉬 모델을 렌더링할 때 미리 생성한 이미지를 적용하여 화면을 생성하는 방법이다. 텍스처 매핑은 미리 생성한 이미지를 다각형 메쉬 모델이 덮어 싸서 마치 모델 표면에 그러한 이미지가 존재하는 것처럼 보이게 하는 것이다.

구현 클래스	클래스 내용
Texture2D	· 2D 텍스처 관련 연산을 위한 클래스 · setImage 메소드 ▷ png 형식의 텍스처 이미지를 로딩하여 각 픽셀의 RGB 값을 정수 배열에 저장한다. · getImage 메소드 ▷ 텍스처 좌표를 인자로 받아서 텍스처 이미지에서 텍스처 좌표에 해당하는 RGB 값을 반환한다.

4. 성능 평가 및 예제

4.1 결과 분석

본 논문의 삼차원 라이브러리를 실제 핸드폰으로 다운로드하여 60초 동안 여러 개의 화면을 생성해 낼 수 있는지 성능을 평가했다.



(a) (b)

(그림 2) 프레임률 검사 예제

● 모델이 화면에 디스플레이 되는 부분이 적은 경우(그림 2(a)) : 삼차원 라이브러리를 이용하여 정육면체로 테스트한 경우

	60초당 프레임 수	프레임/초
wireframe	15	0.25
flat shading	12	0.2
gouraud shading(with texture)	1.2	0.02

● 모델이 화면에 디스플레이 되는 부분이 큰 경우(그림 2(b)) : 위의 모델을 1.5배 확대한 후 테스트한 경우

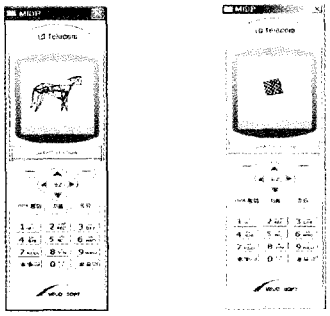
	60초당 프레임 수	프레임/초
wireframe	15	0.25
flat shading	9	0.15
gouraud shading(with texture)	0.42	0.0069

와이어 프레임의 경우는 모델 크기에 상관없이 일정한 프레임률이 나온다. 그러나 쉐이딩이 들어갈 경우에는 픽셀 연산을 해줘야 하므로 모델의 크기에 따라 결과가 많이 달라진다. 특히 텍스쳐 매핑와 그로 쉐이딩 렌더링을 동시에 수행할 경우 속도가 1/3로 줄어들었다. 또한 플랫 쉐이딩 같은 경우도 모델이 더욱 커진다면 그 속도는 훨씬 줄어들 것이다. 왜냐하면 모든 뷰잉 파이프라인을 거쳐서 화면에 픽셀을 찍을 때 RGB값을 계산하

는데 가장 많은 시간이 걸리기 때문이다. 텍스처 매핑을 한 그로 쉐이딩 모드에서는 픽셀 하나의 RGB 값을 결정하기 위해서 깊이, 법선, 텍스처 등을 보강해줘야 하고, 이를 하기 위해서는 많은 소수점 연산을 해야 한다. 하지만 KVM에서는 부동소수점 연산을 지원하지 않기 때문에 LG ez-java MIDP에서 제공하는 MathFP라는 클래스를 이용하였다. MathFP는 정수 형의 변수와 시프트(shift) 연산을 이용하면서 부동 소수점 연산을 해 준다. 실제로 정수 형의 변수를 부동 소수점 연산에 사용하기 위해서는 가상 머신이나 하드웨어에서 지원하는 것보다 불필요한 연산을 수행하기 때문에 많은 시간이 필요하게 된다.

4.2 예제

본 논문에서 구현한 삼차원 라이브러리를 통해 렌더링한 결과는 (그림 3)와 같다. (그림 3(a))는 복잡한 메쉬를 읽어 화면에 나타낸 결과이고, (b)는 메쉬에 텍스처를 입혀 렌더링한 결과이다.



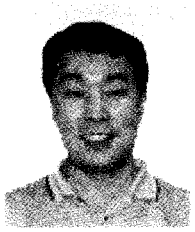
(a) (b)

(그림 3) 삼차원 라이브러리를 사용한 예제

5. 결론 및 향후 연구 방향

본 논문은 모바일 플랫폼 기반 삼차원 어플리케이션 개발에 이용 가능한 삼차원 라이브러리를 개발하였다. 결과적으로 삼차원 모형을 모바일 폰 화면에 나타낼 수 있게 되었고, 와이어 프레임은 물론 플랫 쉐이딩, 그로 쉐이딩, 텍스처 매핑 등 고급 렌더링 기능까지 구현하였다. 성능 평가를 통하여 본 논문의 삼차원 라이브러리를 이용하여 삼차원 렌더링 기능을 제대로 수행함을 보였다. 본 논문의 결과물로서 나온 삼차원 라이브러리는 모바일 플랫폼 기반의 삼차원 어플리케이션에서 이용이 가능할 것이다. 둠이나 퀘이크와 같은 삼차원 게임은 물론 삼차원 실시간 지리 정보 시스템, 아바타 무선 인터넷 채팅 등에 활용될 가능성이 충분하다. 앞으로 삼차원 가속 기능 및 부동 소수점 연산을 지원하는 모바일 폰이 등장한다면, 본 논문에서 개발한 코드를 이용하여 많은 수정없이 훨씬 속도가 빠른 라이브러리 개발이 가능할 것이다.

저자약력



박 곤 호

1984년 조선대학교 전자공학과 (공학사)
1990년 캐나다 토론토대학 데이터통신
1996년 정보통신부 서기관
2001년 한솔 PCS 상무
2002년 (주)위즈케뮤니케이션 사장
현재 (주)일렉트릭아일랜드 사장
관심분야 : 모바일 컨텐츠, 모바일 전자상거래
이 메 일 : khpark@electricisland.com

양 종 열

1993년 경북대학교 컴퓨터공학 (공학사)
1995년 경북대학교 컴퓨터공학 (공학석사)
1997년 포항공대 첨단기술연구소 연구원
2000년 (주)다우기술 대리
현재 (주)일렉트릭아일랜드 개발이사
관심분야 : 모바일 컨텐츠, 자바 가상머신, 영상처리
이 메 일 : jryang@electricisland.com

안 민 수

1999년 포항공과대학교 전자계산학 (공학사)
2001년 포항공과대학교 컴퓨터공학 (공학석사)
현재 포항공과대학교 박사 과정
관심분야 : 컴퓨터 그래픽스, 컴퓨터 애니메이션
이 메 일 : atom@postech.ac.kr

한 우 진

2002년 포항공과대학교 컴퓨터공학 (공학사)
현재 포항공대 석사 과정
관심분야 : 컴퓨터 그래픽스, 컴퓨터 애니메이션
이 메 일 : korbull@postech.ac.kr

이 승 용

1988년 서울대학교 계산통계학 (이학사)
1990년 KAIST 전산학 (이학석사)
1995년 KAIST 전산학 (이학박사)
현재 포항공대 조교수
관심분야 : 컴퓨터 그래픽스, 컴퓨터 애니메이션, 영상처리
이 메 일 : leesy@postech.ac.kr