

3D 게임 엔진 연구: “3D 포트리스” 게임 개발 사례

우석진^o, 이동희, 이동건, 신일경,
박승호, 장호선, 김호준, 최승관, 김경식

호서대학교 게임공학전공

Study on a 3 Dimension Game Engine: Development of the Game “3D Fortress”

Seok-Jin Woo^o · Dong-Hee Lee · Dong-Gun Lee · Il-Koung
Shin · Sung-Ho Park · Hyo-Sun Jang · Ho-Zoon Kim ·
Seung-Kwan Choi · Kyung-Sik Kim

Dept. of Computer Game Engineering in Hoseo Univ.

요 약

온라인 3D 컴퓨터 게임에서 크게 대두되는 두 가지 기술 분야는 3D 연출 기술과 온라인 기술이다. 3D 연출 기술은 3D 가속기의 출현으로 더욱 발전을 할 수 있었고 MicroSoft사의 DirectX가 나오므로써 다양한 하드웨어를 좀더 쉽게 제어할 수 있게 되어 게임 산업은 한층 더 발전하게 되었다. 그러나 DirectX기반의 3D 기술을 응용하여 게임 엔진을 만드는 데는 게임의 특징에 따라 엔진 구성이 크게 차이가 나기 때문에 최종 게임 엔진까지는 사용자가 직접 만들어야 하는 어려움이 있다. 본 연구에서는 DirectX를 이용해서 3D 온라인 기반의 게임을 제작할 수 있는 3D 엔진을 구현하였고 포트리스 3D 라는 3차원 온라인 슈팅 게임의 개발에 적용한 사례를 발표한다.

1. 서론

3D 게임을 만들기 위해서는 3D 연출을 쉽게 하기 위한 3D게임엔진이 필요하다. 3D 게임 엔진은 3D Object를 데이터 베이스에

서 Load시켜 조작하고 실시간으로 연출하기 위한 함수들과 데이터 구조의 집합이라고 할 수 있다. 물론 조명, 재질, 변형, 애니메이션, 3D 클리핑, 특수효과 등의 다양한 추가적인 기능을 지원해야 할 경우도 있다. 이러한 연출을 위해 윈도우 환경의 컴퓨터에서 동작하는 월드 클래스 개념이나 3D 멀티미디어 등

을 쉽게 만들도록 해주는 삼차원 그래픽 솔루션이 바로 Direct3D 이다.

Direct3D의 주된 역할은 장치 독립적인 방식으로 3D 비디오 디스플레이 하드웨어를 위한 드로잉 인터페이스라고 할 수 있겠다. 그런데 Direct3D에서는 보호모드와 직접모드의 두 가지 모드가 제공되며 보호모드는 애플리케이션을 빠르게 개발하기를 원하는 개발자에게 애니메이션 체계 등을 사용하는 매우 편리한 상위 레벨의 API 인터페이스를 제공한다. 이와 반대로 즉시모드는 성능위주의 프로그래밍을 위한 하위 레벨 3D API 인터페이스이다. 따라서 즉시모드는 게임 분야나 또는 높은 수행 속도, 성능을 필요로 하는 애플리케이션을 제작하기 위해 사용되는 이상적인 솔루션을 제공한다.

본 연구에서는 3D Object data를 얻기 위해서 3D Studio MAX에서 제공하는 ASE file을 사용하여 이를 Load하고 조작하는 기술을 알아보고, Direct3D를 이용한 3D게임 엔진을 연구하여 이를 응용한 게임을 소개하고자 한다.

2. 3D 엔진의 구성 및 기능

3D 엔진은 3D 데이터 적재모듈과 Object 연출부분으로 나눌 수 있는데, 연출부분에서는 연산, 투영, 클리핑, 변형, 스케일링, 렌더링 등의 요소가 필요하다

이를 처리하기 위한 데이터 구조는 ASE파일의 Parsing을 통해 재질 정보와 3D 지오메트리(geometry) 정보들을 추출하여 Object 구조체 데이터를 만들어야 한다 (그림 1).

```

<Material 구조체>
float          ar, ag, ab;
float          dr, dg, db;
float          sr, sg, sb;
int            two_sided;
float          self_illum;
float          transparency;
float          shine;
float          shine_strength;
float          uvw_u_tiling;
float          uvw_v_tiling;
char          material_name[20];
int            diffuse_id;
int            diffuse_used;
int            diffuse_amount;
char          d_material_file_name[128];
LPDIRECTDRAW7 d_surface;

int            opacity_id;
int            opacity_used;
int            opacity_amount;
    
```

```

char          o_material_file_name[128];
LPDIRECTDRAW7 o_surface;

<ASE 파일 구조체>
int            vertex_num;
int            face_num;
int            material_num;
D3DVERTEX     *vertex_list;
T_MATERIAL     *material;

<Object 구조체>
T_ASE          *ase;
D3DVECTOR     loc;
D3DVECTOR     angle;
D3DVECTOR     scale;
    
```

그림 1. 구조체 데이터

3D 엔진의 기능을 살펴보면

- 모든 종류의 텍스처 Wrapping지원(Plane, Box, Cylinder, Sphere)
- 먼 단위 Texture Wrapping지원
- RGB color rendering
- Diffuse/Opacity Texture mapping
- Self-illumination
- Specular light
- Transparency
- Alpha blending
- Ambient, Omni, Spot, Point, Directional Light 기능 및 표면에 실시간 광선 합성으로 light 애니메이션
- Library사용법의 경량화
- Draw와 3D의 통합형 엔진 구조
- 내부 리스트 사용으로 편리한 restore기능 및 메모리 관리

3. 지오메트리 파이프라인

지오메트리 파이프라인은 입력으로 정점들을 받아들이며 변환 엔진은 입력된 정점들을 위해 세 개의 변환(월드, 뷰, 프로젝션)을 적용시킨다. 또한 여기에서 나온 결과를 클립하고 모든 것들을 주사로 전달한다. 일반적인 파이프라인의 구조는 그림 2와 같다.

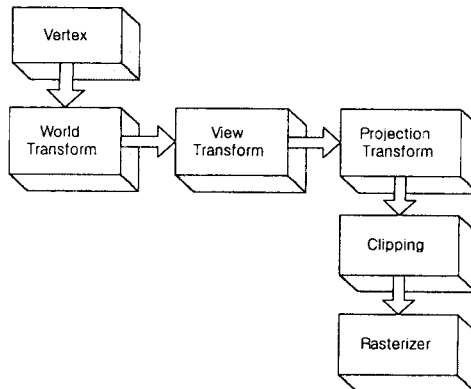


그림 2. 일반적인 파이프라인

월드 변환에서는 모델의 실세계 좌표로의 변환을 수행한다. 바로 이 과정에서 이동, 회전, 스케일 변환이 수행되어진다. 뷰 변환에서는 앞에서 처리된 정점들을 카메라 공간에 배치시킨다. 그리고 프로젝션 변환에서는 카메라 근처에 있는 객체가 멀리 떨어져 있는 객체보다 더 크게 팽창되어 보여지는 효과를 만든다. 그 다음에 너무 가까이 있는 것과 너무 멀리 떨어진 객체들이나 화면에 보이지 않는 객체들을 그리지 않는 클리핑과정이 이루어진다. 마지막으로 화면에 보이는 객체들을 래스터라이저에게 전달하고 래스터라이저는 정점들을 스크린 좌표로(X, Y) 변환하여 그림을 그리게 되고, 이때 Z값으로 깊이 버퍼 같은 것을 사용해서 사실적인 효과를 연출할 수도 있다. 엔진을 설계하는 과정에서 이러한 지오메트리 파이프라인의 설계에 따라 엔진의 성능이 크게 달라지게 된다. 객체의 모든 정점들마다 각자의 재질 정보가 존재하는데 재질 정보를 매번 변경하게 내부 설정 변경 때문에 엔진에 속도 저하효과를 가져오게 된다. 그래서 이번 엔진에서는 이러한 문제를 해결하기 위해서 객체의 모든 재질 정보를 먼저 파악하고 그 정보를 토대로 3D 데이터를 재구성하여 재질 정보에 따라 객체의 폴리곤을 그룹화 함으로써 이러한 재질 정보 변경을 최소화 할 수 있었고, 이로 인하여 커다란 속도 향상을 이룰 수 있었다. 그리고 특수효과 연출부분에서 아주 많이 사용되는 Alpha Blending 기법의 사용은 파이프라인에 약간의 변화를 주었다.

Alpha Blending을 처리해야하는 정점 데이터들은 일반적인 파이프라인의 구조로는 올바른 연출 효과를 연출 할 수가 없다. 그래서 정점 그룹 처리과정에서 Alpha Blending효과가 발생하면 이 정점그룹을 큐(Queue)에 담아 두었다가 일반 정점의 처리가 모두 끝난 후에 큐 안에 들어 있던 정점들을 처리하는 구조로 이러한 문제를 해결하여 올바른 연출을 할 수 있었다. 파이프 구조는 그림 3과 같다.

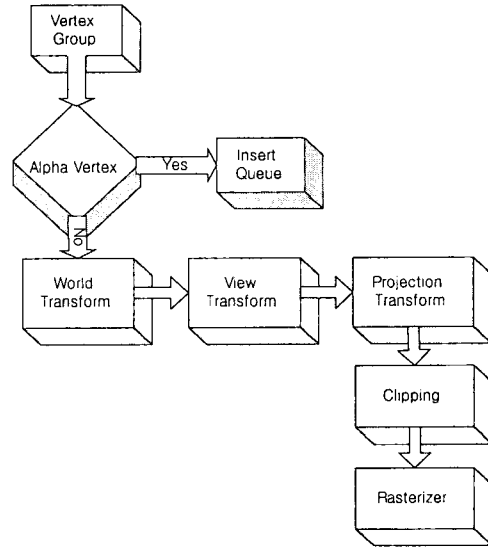


그림 3. Alpha Blending처리 파이프라인

4. 3D 엔진을 이용한 게임 구현

1) 게임 개요 및 기획

- (1) 장르는 3D 온라인 슈팅게임으로 분류한다.
- (2) 게임 유닛은 탱크를 기본으로 한다.
- (3) 맵의 구성은 평면의 맵을 기본으로 한다. 배경으로는 반구를 덮어서 보여준다.
- (4) 3D 효과를 살리기 위해 다양한 카메라 시점의 효과를 넣는다.
- (5) 상대방의 탱크를 맞추기 위해서는 우선 탱크의 방향과 포대의 각도를 생각하여 적당한 힘으로 폭탄을 발사해서 상대방을 맞추게 된다.
- (6) 카메라시점에 많은 비중을 두어 여러 가지 카메라 시점을 만든다.

2) 게임의 흐름도

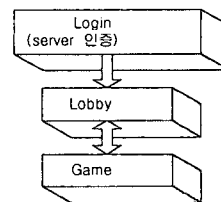


그림 4. 게임 흐름도

본 게임은 온라인 게임이기 때문에 게임을 하기 위해서는 먼저 서버에 접속을 하여 자신의 ID를 인증받아야 한다. 인증을 받은 후에 게임을 하기 위해 모여 있는 Lobby로 가게 된다. 거기서 다른 User 와 대화를 하면서 다른 User 가 만들어놓은 방으로 접속을 하여 게임을 하게 되고 아니면 자신이 직접 게임의 Host 가 되어 게임을 준비할 수 있다 (그림 4 참고).

3) 게임 화면의 구성

(1) 게임 Login 화면

게임을 하기 위해 서버의 인증을 받는 화면을 그림 5에 보인다.



그림 5. 게임 Login 화면

(2) 게임 Lobby

게임을 하기 전에 잠시 대기하는 곳으로 이곳에서 만들어진 게임에 접속을 하거나 자신이 직접 방을 만들 수 있다. 그리고 다른 User 들과 여러 가지 정보를 주고받기도 한다.

(3) 방 만들기

자신이 직접 방을 만들기 위해서 필요한 정보를 입력한다.

(4) 게임 준비

방을 접속을 한 후에 바로 게임이 시작되는 것이 아니고 이곳에 대기하면서 게임에 참여하는 User 수가 적당히 참여했을 경우 게임

을 시작하게 된다.

(5) 게임 화면

게임의 통신은 Peer - to - Peer 방식으로 하지만 Server와의 접속은 해제하지 않는다 (그림 6 참고).

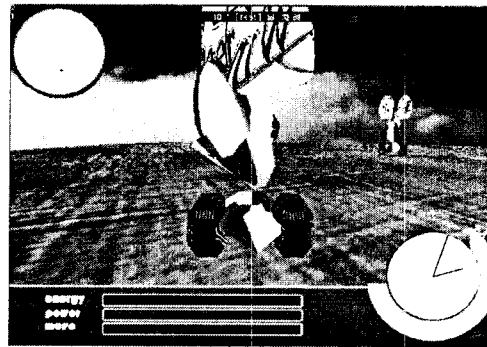


그림 6. 게임 화면

6. 결론

이상에서 3차원 게임을 만들기 위한 엔진 기능과 지오메트리 파이프라인에 대해서 살펴보고, 이를 이용한 간단한 3D 게임을 제작해보았다. 물론 이 엔진을 활용하여 다양한 형태의 3D 게임을 제작할 수 있으며, Network 속도의 개선과 3D 엔진의 발전이 이루어진다면 좀더 현실적이고 Interactive한 게임을 만들 수 있다. 앞으로 3D 효과에 단순히 카메라 움직임뿐만 아니라 다양한 애니메이션 기능, 빠른 실시간 충돌검사, 여러 가지 특수 효과 등을 접목하여 보다 더 발전된 형태의 게임을 제작 할 수 있는 엔진으로 발전시킬 것이다.

참고 문헌

- [1] LaMothe 著, 김숙자 譯, "Black art of 3D game programming," 성안당, pp. 705-706, 1996. 4.
- [2] 안승중 著, "Direct 3D contact" 대림, pp.409-432, 2000. 6.
- [3] 김영환 著, "Direct 3D programming", 테크&북, pp.538-586, 1999.9.
- [4] Micro soft, Direct X 7.0 SDK 3DIM Document file.