저사양 PC를 위한 1인칭 슈팅게임

안성옥*, 김수균*

요약

대부분의 1인칭 슈팅게임은 게임엔진을 이용하여 제작되고 있으며 기본적으로 고기능의 충돌감지 기능을 제공한다. 충돌감지는 게임제작에서 매우 중요한 역할을 한다. 본 논문에서는 게임엔진을 사용하지 않고 전 과정을 DirectX를 이용하여 1인칭 슈팅(First-Person Shooter:FPS) 게임을 제작하는 방법에 대해 제안한다. 기존 고기능 게임 엔진에서 제공하고 있는 충돌 검출 기능을 사용하지 않고 DirectX 라이브러리를 이용한 방법으로 저수준 PC에서도 사용할 수 있는 장점과 메모리 측면에서 효율적인 장점이 있다. 또한 객체를 감싸는 구들의 반지름만 비교함으로써 객체의 충돌 여부를 판정하는 방식을 통하여 효율적인 경계구 충돌 검출을 이용하여 충돌 감지를 수행한다.

First Person Shooter game for low-end PC

Seong-Og An*, Soo-Kyun Kim*

ABSTRACT

Most first person shooter games are produced using game engines and are fundamentally equipped with high quality collision sensing functions. This function of detecting collisions plays an important role in game production. This method proposes a method of making FPS games not through game engines but through DirectX. The advantage of using DirectX instead of collisions detecting functions of game engines is that one can use it in low-end PC and it is efficient in terms of memory. Also, the method of DirectX can proficiently sense collisions by comparing the radian of the spheres that surrounds the objects which will judge whether there was an impact between the objects.

Key Words : Collision detection, bounding sphere, distance, FPS, Low-end PC

* 배재대학교 게임공학과(✉sungohk@pcu.ac.kr)
· 제1저자(First Author) : 안성옥 · 교신저자(Correspondent Author) : 김수균
· 접수일(2010년 5월 25일), 수정일(1차 : 2010년 6월 30일), 게재확정일(2010년 7월 5일)

Ⅰ. 서 론

현재 게임시장에서 1인칭 슈팅게임 게임 수가 급격하게 늘고 있다. 특히 온라인 게임의 전유물이라고 여겼던 MMORPG에서 벗어나 FPS 게임이 시장에서 큰 성공을 이뤘다. 현재 국내에서 가장 큰 인기를 모으고 있는 게임으로는 서든어택이 있으며, 이러한 게임들은 게임엔진을 이용하여 제작한다. 게임엔진을 사용하면 기본적으로 모듈화 된 고기능의 자원들을 쓸 수 있지만 컴퓨터 사양이 높아진다는 단점이 있다. 제안방법은 고기능의 게임엔진을 사용하지 않고 저수준 PC에서도 동작할 수 있도록 DirectX 라이브러리를 이용하여 게임을 제작하고, 효과적인 충돌 감지를 위해 경계구 충돌(Bounding Sphere) 방법을 사용한다. 이는 객체를 감싸는 구들의 반지름을 비교함으로써 객체와 객체의 충돌 여부를 판정하는 방식으로 매우 빠르게 충돌을 검출 할 수 있는 장점이 있다[1,2].

Ⅱ. 연구 배경

국내 많은 게임 사용자들의 대부분은 MMORPG와 FPS 게임에 몰려 있다고 볼 수 있다. 이렇게 많은 사용자층을 가지고 있는 두 개의 장르 중에서 본 논문은 DirectX를 이용하여 3차원 FPS게임을 구현한다. 기존에 상용화 되어 있는 게임들은 고기능의 게임엔진을 사용하여 화려하고 복잡한 기능을 통해 사용자에게 큰 재미와 볼거리를 제공하고 있다. 그러나 제안방법은 고기능의 게임엔진을 사용하지 않고 DirectX 라이브러리를 이용하여 게임을 개발한다. 게임엔진만을 이용하여 게임을 제작하게 되면 모듈화 된 리소스를 사용하여 편리하다는 장점이 있는 반면에 세세한 부분까지 컨트롤이 어렵다는 단점이 있다. 하지만 DirectX 라이브러리를 이용하면 게임엔진으로는 접근할 수 없는 세세한 부분까지도 구현 가능하다는 큰 장점이 있기 때문에 본 논문에서 이를 중심적으로 연구한다.

Ⅲ. FPS 게임의 설계

3.1 전체적인 게임 시스템의 개요

그림 1은 전체적인 게임 시스템의 개요도이며 일반적인 게임 루프이다. CPU는 사용자의 입력과 게임로직, 사운드를 반복적으로 처리하게 되며 이것은 하드웨어에서 실행되도록 최적화 되었다. 추가적으로 사용자 입력은 키보드나 마우스의 움직임에서 핵심 프레스를 받아들이는 과정이라 할 수 있다.

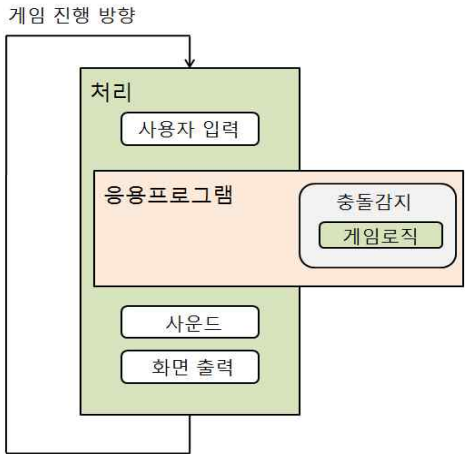

그림 1. 시스템 개요
Fig 1. System flowchart

Ⅳ. 게임의 구현

본 절은 DirectX를 이용한 FPS 게임이 구현되는 부분에 대한 내용으로 게임에 사용된 기술 구현에 대해 중심적으로 다루게 될 것이며 또한 지형과 입자 총탄 그리고 충돌감지 구현 방법에 대해서 설명 할 것이다.

4.1 지형 생성

그림 2는 사실적인 지형을 생성한 화면으로 지형에 텍스처를 입혀 실사와 비슷한 지형을 구현한다.

그림 2. 지형모델생성
Fig 2. Generation of terrain model

사실적인 지형생성 후 지형에 카메라를 이식하여 지형 위를 걷거나 뛰는 효과를 구현한다. 이러한 지형생성의 장점은 산이나 계곡을 표현할 때 사실적으로 표현 할 수 있다는 장점과 텍스처에 따라 해변이나 풀 덮인 언덕, 눈 덮인 산 등을 연출할 때 매우 효과적이라는 장점이 있다.

이 기술을 사용하려면 특정 버텍스의 높이와 대응되는 배열을 이용할 수 있으며 이것은 그레이스케일 즉, 회색 톤으로만 구성된 raw파일로 구현하며 회색 톤의 밝기 음영에 따라 높이 맵 각각의 요소에 대응된다. 따라서 지형 격자 내의 각 버텍스와 일대일 대응관계를 가지는 행렬로 생각할 수 있다.

4.2 입자 총탄

본 절은 입자총탄 구현부분으로 입자총탄은 파티클 시스템을 사용하여 개발한다. 파티클 시스템은 신비로운 자연 현상의 많은 부분을 비슷하게 구현 가능하고 무수히 작은 입자들로 표현가능 하다. 파티클 입자들은 눈이 내리는 장면을 연출 하거나 기관총에서 발사되는 총알과 같은 것들을 모델링한다. 파티클 시스템에서 기본의 되는 것은 원하는 개수와 텍스쳐 파일을 초기화하는 것이다. 매 프레임 업데이트를 통해 새로운 좌표를 계산하여 렌더링 한다.

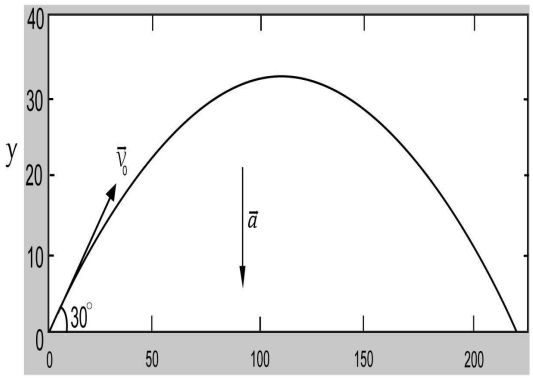

그림 3. 가속도에 따른 중력
Fig 3. Acceleration gravity

그림 3은 입자 총탄이 발사될 때 총알처럼 빠른 속도로 움직이는 가속도에 따른 중력의 변화 그림이다. 가속도에 따라서 중력의 값을 지정해 줌으로서 중력의 영향을 받고 진짜 총알이 발사되는 효과를 구현한다.

마지막으로 사용자는 키 입력을 통해 카메라 위치에서 카메라가가 바라보는 방향으로 발사되는 입자 총탄을 모델링 한다.

4.3 충돌감지

충돌감지는 두 물체가 서로 맞부딪치거나 움직이는 물체를 접촉할시 사물의 상태를 검사하는 것이다. FPS 게임을 구현하기 위해서는 필수 요소이며 가장 중요한 부분이다.

충돌감지를 통해서 게임세계를 현 세계와 동일한 것을 개발할 수 있으며 많은 것을 이룰 수가 있다. 사람이 건물 사이로 지나가는 것을 방지하기 위해서도 충돌감지가 필요하며 총알과 적군 사이에 충돌감지를 통해 게임의 완성도를 높이는데 크게 작용한다.

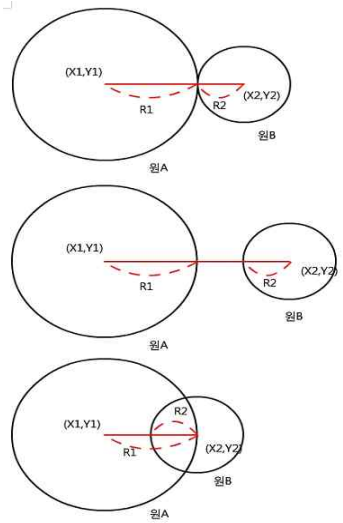

그림 4 경계구체를 이용한 알고리즘
Fig 4. Algorithm using bounding sphere

그림 4은 경계구 볼륨을 이용한 알고리즘으로 본 논문에 제시하는 방법으로 원 A의 반지름 R_1과 원 B의 반지름 R_2의 거리의 사이를 이용하여 원A와 원B가 동일선상에 있을 때와 겹쳤을 시 충돌감지를 한다.

식(1)의 d는 원A의 반지름과 원B의 반지름의 합을 나타낸 수식이다.

$$d = R_1 + R_2 \qquad \text{(식1)}$$

아래의 식(2)은 그림 4 경계구를 이용한 알고리즘으로 풀어쓴 수식이며 경계구 충돌감지 알고리즘을 적용하기 위해 사용되는 두 구체의 반지름 합의 거리를 계산하는 수식이다.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \qquad \text{(식2)}$$

FPS 게임에서 원A는 카메라(사용자시점)이며 원B는 충돌대상(객체)이다. 본 절에 적용하기 위해 경계구 볼륨과 카메라의 충돌감지를 하기 위해서는 위의 (식2)을 대입하여 카메라와 경계구의 충돌감지를 구현한다. 충돌감지가 발생되면 카메라의 속도 값은 Z축으로 (-)속도 값을 주며 카메라가 움직이지 않도록 구현하였으며 게임화면시 건물을 통과하지 못하는 충돌감지가 발생한다.

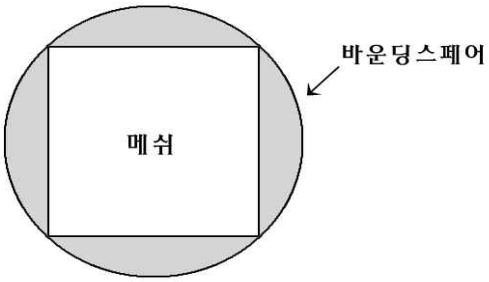

그림 5. 바운딩스페어와 메쉬관계1
Fig 5. Bounding sphere and mesh1

그림 5는 바운딩스페어 안에 포함되는 메쉬 사이에 빈 공간이 발생되며 이 빈공간은 충돌감지의 오차범위이다. 즉 그림과 같이 완벽한 충돌감지가 구현되지 않는 것이다.

이 방법을 해결하기 위해 그림 6과 같이 메쉬 안에 새로운 메쉬를 생성하여 바운딩 스페어를 생성함으로서 오차 범위를 줄였다.

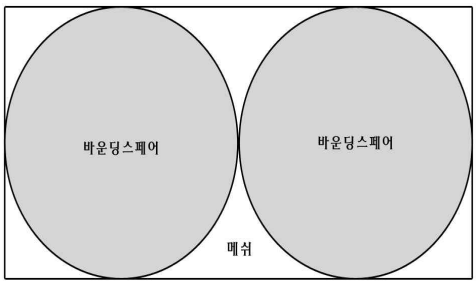

그림 6. 바운딩스페어와 메쉬관계2
Fig 6. Bounding sphere and mesh2

이후 메쉬는 덮어쓰는 방법을 택하여 보다 오차범위가 적고 완벽한 충돌감지를 구현한다. 또한 총알과적(표적) 충돌감지 구현에는 위의 알고리즘을 사용하여 적용하였다. 카메라를 이용한 것과는 달리 파티클(총알)을 사용하여 충돌감지를 구현한다.

V. 게임화면

본 논문에서 구현된 개발환경은 다음과 같다. 펜티엄4 2.6GHz, 512MB램, 윈도우 XP환경에서 Visual Studio2008와 DirectX 라이브러리를 이용하여 수행하였다.

그림 7. 타겟을 겨냥하고 있는 화면
Fig 7. Targeting

그림 8. 입자총탄을 발사한 화면
Fig 8. Open fire

그림 7과 8은 게임을 플레이 하는 화면으로써 목표를 겨냥하고 입자총탄이 발사되는 모습을 나타낸다.

그림 9. 건물 충돌 감지하는 화면
Figure 9. Collsion detection

그림 9는 충돌 감지를 통하여 건물을 통과하지 못하는 모습을 나타낸다.

VI. 결론

본 논문은 DirectX 라이브러리를 이용하여 1인칭 슈팅 게임을 제작함으로서 기존의 게임엔진에서 제공하고 있는 충돌 검출 기능을 사용하지 않고 저수준 PC에서 동작 가능한 게임을 제작하였다. DirectX 라이브러리를 이용하면 게임엔진으로는 접근 할 수 없는 세세한 부분까지도 구현 가능하다는 점과 객체를 감싸는 경계구들의 반지름의 거리를 계산함으로서 객체의 충돌 감지가 발생하는 방식을 통하여 효율적으로 경계구 충돌 감지를 하는 장점이 있다. 이러한 장점들로 인해 저사양 PC에서 활용 가능한 1인칭 슈팅게임이라고 할 수 있다

Acknowledgement

"본 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2010-0015563)"

안성옥(Song-Og An)

1983 고려대학교 수학교육과(이학사)
1985 고려대학교 컴퓨터학과(이학석사)
1989 고려대학교 컴퓨터학과(이학박사)
1991년~현재 배재대학교 게임공학과 교수
※ 관심분야 : 멀티미디어시스템, 데이터베이스, 게임개발

김수균(Soo-Kyun Kim)

2006년 고려대학교 컴퓨터학과 (이학박사)
2006.3 ~2008.2 삼성전자 통신연구소 책임연구원
2008년~현재 배재대학교 게임공학과 조교수
※ 관심분야 : 기하모델링, 게임그래픽, 실감미디어