논문 03-02-17

3D 충돌 검출 모델의 선정 기준에 관한 연구

Study on the Selection Criteria of 3D Collision Detection Model

姜 允 美*, 朴 容 範*
Yun-Mi Kang*, Young B. Park*

요 약

좋은 3D 엔진이란 객체들의 상호 작용을 실세계와 유사하게 표현하는 물리학 엔진을 말한다. 충돌은 이런 상호작용 중의 하나이며, 충돌 유무 검사와, 충돌 지점, 충돌후의 반응을 다룬다. 대부분의 물리학 엔진과 같이 충돌 검사도 정확하게 검출하려면 많은 시간이 소요된다. 그러나 요구되는 정밀도만 만족하는 모델 사용으로 검출 시간을 조절 할 수 있다. 그러므로 요구되는 정밀도와 시간에 따라 충돌 검출 모델을 선정하여야 한다. 본 논문에서는 충돌 검출 모델들을 정확도에 따라 7단계로 구분하여 구현하고 실험했다. 이를 통해 시간과 정확도의 연관 관계를 분석하고, 이에 따른 충돌 검출 모델을 선택하기 위한 기준을 제시하고자 한다.

Abstract

In a good 3D engine, objects interactions are similar to those of real-world. Collision is one of the interactions. It includes whether collision took place or not, where collision took placed, and reaction after collision took place. More precise collision detection needs more time. If there exist required precision, detection time can be controlled by choosing appropriate detection model. Therefore, we need a selection mechanism for the collision detection with respect to required precision and detection time. In this paper, a collision detection model with seven different precision levels is examined. And relationship between detection time and precision is analyzed. Consequently, we propose a selection mechanism for collision detection model.

keyword : 3D, engine, collision, detection, model

1. 서 론

현실적인 3D 영상의 표현에 있어서 실시간적인 물리학 엔진은 3D 영상 세계의 객체들 사이의 실제적인 상호 작용을 제공한다. 이러한 물리학 엔진은 관찰자에게 좀 더 큰 현실감을 제공하며, 관찰자는 더욱더 흥미로운 영상을 즐길 수 있다.[5] 하지만, 현실감 있는 엔진은 상당히 많은 자원을 요구하기 때문에 아무리 성능이 좋은 컴퓨터일지라도 현실적으로 3D 영상에 적용하기는 어렵다. 이러한 문제점 때문에 개발자들은 한정된 자원에서 최적의 속도를 낼 수 있는 엔진

*檀國大學校 電子計算學科
(Dept of Computer Science, Dan-Kook Univ.)
※본 연구는 2003학년도 단국대학교 교내 연구비의 지원으로 이루어졌습니다
接受日:2003年 8月 22日, 修正完了日:2003年 11月 18日

을 개발하기 위해 노력한다.[6][10]

충돌 검출은 실시간 물리학 엔진의 중요한 부분이다[5]. 3D 영상에서 충돌 검출은 매 프레임마다 수행되며, 충돌의 유무를 검사하고, 충돌한 지점, 충돌 후의 반응을 다룬다.[2]

현재 사용되는 충돌 검출법은 점(vertex)과 점, 점과 선(edge), 점과 면(face), 면과 선, 구(sphere)와 선, 구와 면, 상자(box)와 선, 상자와 면, 상자와 상자, 구와 구 등이 있다.[1][7][11] 3D 영상의 종류나 각 상황마다 그 효율성이 다르기 때문에[10] 어느 방법이 더 우수하다고 말하기는 어렵다.

기존 연구에는 충돌 검출에 관한 여러 알고리즘이 소개 되었으나, 어떤 상황에 어떤 알고리즘이 적용되어야 하는지에 관한 연구와 하나의 알고리즘이 상황이 다른 각각의 영상에 어떠한 방법으로 적용되어야 하는지에 관한 연구는 진행되지 않고 있다. 본 논문에서는 하나의 알고리즘을 각 영상의 상황에 맞게 효율적으로 적용하는 방법을 소개하고자 한다. 먼저 기존에 사용되고 있는 충돌 검출법 중에서 점(vertex)과 면(face)의 충돌 검출 기반의 상자(box)충돌기법 중 가장 널리 사용되는 경계 상자기법을 소개하고, 정밀도에 따른 충돌 검출시간을 분석한다. 이를 통하여 적합한 충돌 검출 방법의 선정기준을 제시하려 한다.

2. 충돌 검출법과 특징

대부분의 충돌 검출 모델의 특징을 살펴보면 속도와 정밀도는 반비례 관계에 있으며, 3D 영상과 객체의 특징에 따라 적용되는 모델이 다르게 사용된다. 따라서 개발자는 시간과 속도를 고려하여 각 상황에 적합한 충돌 검출법을 선택 또는 개발해야 한다.[9]

일반적으로 충돌 검출 계산은 영상이 2D이냐 3D이냐에 따라 두 종류로 나뉘고, 움직이는 객체와 환경을 구성하는 정적인 기하구조 사이의 충돌 검출인지, 아니면 움직이는 객체들 사이의 충돌 검출인지에 따라 두 종류로 나뉜다.[6]

특히 3D에서는 얼마만큼의 정확한 충돌 검출이 요구되는지에 따라 적절한 방법이 소개되어야 한다. 예를 들어 두 캐릭터가 칼싸움을 한다면, 칼이 충돌된 지점에 상처가 나게 해야 하므로 정확한 충돌 검사가 필요하다. 반면 미사일을 떨어뜨린 지점의 폭파장면과 같은 경우는 정확한 충돌 지점보다는 충돌의 유무만 파악하고 폭파장면을 연출하므로 낮은 정확도로 충돌 검사가 가능하다.

3D 충돌 검출에서는 객체의 세세한 부분을 감싸는 객체의 경계(bound)로 경계 볼륨을 만들어 사용한다.

2.1 경계 볼륨

3D에서 객체를 표현하는 것은 다각형(polygon)이며, 폴리곤을 구성하는 것은 하나하나의 픽셀이다. 완벽한 충돌 검출을 하려면 각각의 폴리곤, 또는 폴리곤을 구성하는 픽셀들의 위치를 계산해야 하지만, 한정된 컴퓨터 자원 때문에 많은 시간을 소비하기는 어렵다. 그래서 각 객체를 감싸는 최대한 작은 볼륨을 만드는데, 객체를 구성하는 볼륨이 구 형태이면 BS(Bounding Sphere: 경계구)라 하고, 육면체 모양이면 OBB(Oriented Bounding Box: 경계상자)라고 한다. 특히 육면체 모양의 경계상자가 축에 평행한 경우를 AABB(Axis-Aligned Bounding Box: 축에 정렬된 경계상자)라고 한다.[3] 이 외에도 원기둥 모양, 원뿔 모양과 같이 특이한 모양으로 볼륨을 형성할 수 있다.

그림 1. 원기둥 모양의 볼륨으로 표현된 객체
Fig. 1. a cylinder model for the collision detection

그림 1.은 움직이는 캐릭터를 원기둥 모양을 볼륨으로 표현한 것이다. 즉 본 논문에서의 볼륨은 객체와 동일하며, 볼륨 수는 객체 수를 말한다.

BS는 아케이드 스타일의 우주 전투기 게임과 같은 3D 영상에 적합하며, 충돌 검출 방법이 간단하고 구현도 쉽기 때문에 2D나 3D에서 많이 사용된다. AABB는 game world의 축과 평행한 정육면체 형태이기 때문에 볼륨을 형성하는 시간을 절약한다. OBB는 game world 축에 정렬되지 않은 육면체 형태이며 볼륨을 형성하는데 AABB보다 많은 연산을 필요로 한다. 하지

만 객체에 가장 잘 맞는 경계상자를 만들 수 있고 충돌 검출 결과가 AABB보다 정확하다.[4]

세밀한 충돌 검출을 위하여 하나의 객체를 여러 부분으로 분할하여 각 부분을 하나의 볼륨으로 구성하는 방법이 있다[1]. 사람 캐릭터를 예로 들면, 서 있는 사람이라면 전체를 직육면체로 표현할 수 있다. 하지만 팔 다리가 움직이고, 정확한 충돌 검출이 필요한 경우라면 사람 객체를 머리, 몸통, 팔, 다리 로 분할한다.[1][8] 이때 머리는 BS 또는 AABB, 몸통은 AABB, 팔과 다리는 OBB로 표현한다. 좀더 세밀한 충돌 검출을 원한다면 팔을 윗부분과 아랫부분으로 다시 나눌 수 있다. 이렇게 분할한 것을 하나의 트리로 구성하여 이런 볼륨들이 사람이라는 객체를 구성한다.[9]

충돌 검출 시간을 줄이기 위한 전략중 하나로 계통적으로 범위를 좁혀나가면서 충돌을 검출하는 방법이 있다.[5] 움직이는 사람일 경우 사람객체의 BS를 먼저 만들어 충돌 검사를 한다. 충돌이면 좀더 세밀한 검사를 위해서 볼륨을 OBB 또는 AABB로 하고 계산한다. 이 방법은 모든 객체에 정밀한 충돌 검사를 하는 것이 아니라, 세밀한 검사를 할 필요가 있는 객체를 선별하는 경우에 적합하다.

2.2 두 경계 상자의 충돌

OBB는 객체의 중심으로 좌표계를 생성하므로, 모든 OBB들은 고유의 좌표계를 갖는다. 따라서 두 OBB의 충돌을 검사하기 위해서는 서로 다른 두개의 좌표계를 동일시켜야 한다.[2] 그림 2은 로컬 좌표계와 월드 좌표계의 변환을 보여준다.

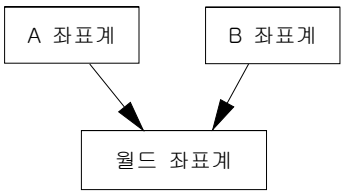

그림 2. 로컬 좌표계와 월드 좌표계

Fig. 2. Local coordinates system and World coordinates system

경계 상자의 충돌 검사는 위와 같은 좌표계 변환 작업을 거친 후에 이루어진다.

그림 3는 면과 점의 교차 판정 이론에 근거한 상자 충돌 검사방법을 보여준다.[11][12][13] 이것은 OBB B의 꼭짓점 V가 OBB A의 한 평면의 위에 있는지 아래에 있는지 또는 평면상에 있는지를 구분하여 평면 위에 있을 때는 외부에 있다고 판정하고 평면 아래에 있을 때는 내부에, 그리고 평면상에 있으면 접해있다고 판정한다.

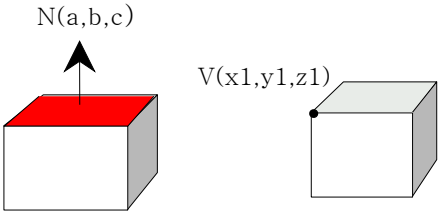

그림 3. OBB A 와 OBB B

Fig. 3. OBB A and OBB B

먼저 $N=(a,b,c)$ 가 OBB A의 면과 수직으로 만나고 상자의 외부로 향하는 방향벡터(face normal vector)라 하고 평면위의 한점 $P=(x',y',z')$ 라 하면 평면의 방정식은 식 (2.1)과 같다.

$$ax+by+cz+d=0 \tag{2.1}$$

여기서 $d=-(ax'+by'+cz')$ 이다. 식 (2.1)에 OBB B의 꼭지점 $V=(x_1, y_1, z_1)$ 를 대입하자. V 가 평면에 접해 있을 경우 0이 되며, 외부에 존재하면 양수 값을 가지게 되고 내부에 존재하면 음수의 값을 가진다.

$$\begin{aligned} &a x_1+b y_1+c z_1+d>0 \ : \text{외부} \\ &a x_1+b y_1+c z_1+d<0 \ : \text{내부} \\ &a x_1+b y_1+c z_1+d=0 \ : \text{접함} \end{aligned} \tag{2.2}$$

이제 그림 3의 OBB B의 꼭지점 위치는 식 (2.2)로 판별이 가능하다.

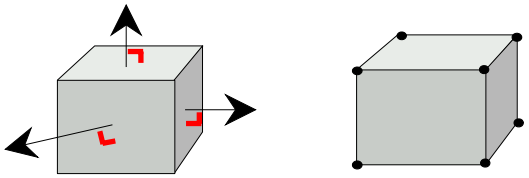

그림 4. OBB A와 OBB B의 충돌 검출
Fig. 4. Collision Detection between OBB A and OBB B

그림 4의 OBB A를 구성하는 6개 평면의 방향벡터(face normal vector)를 구하고 OBB B의 꼭지점 위치를 파악한다. 만일 꼭지점이 6개 평면모두에 대해 내부에 위치한다고 판정이 되면 이것은 이 점이 상자에 내부에 있음을 의미한다. OBB B의 8개 꼭짓점 모두에 대해 같은 방법을 적용하여 그 중 하나라도 OBB의 내부에 존재하면 충돌로 판정한다.

이 방법은 면과 점 사이의 교차여부를 판정하므로 속도는 빠르지만 선과 선, 면과 선의 충돌 여부를 판단하는 데 많은 문제점을 가진다. 예를 들어 그림 3과 같은 경우에 정확한 충돌을 알 수 없다. 이러한 상황은 긴 칼로 긴 나무를 베는 것과 같은 경우에 발생한다.

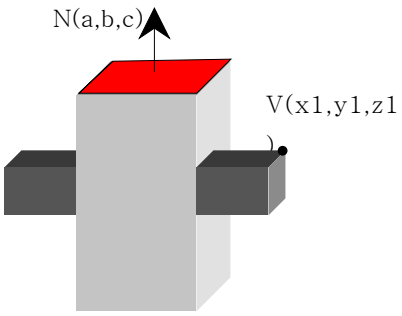

그림 5. 긴 상자의 충돌
Fig. 5. Collision of long boxes

일반적으로 움직이는 3D 물체의 충돌은 움직임이 충분히 느릴 경우 그림 5의 상황으로 바로 변하지 않고 한 상자의 꼭지점이 다른 상자의 내부를 지나 통과하여 그림 5과 같이 변화한다. 하지만 움직임이 빠르면 꼭지점이 통과하는 순간을 지나치게 되어 충돌 감지에 실패한다.

3. 사이점을 이용한 충돌 검출 방법

앞에서 소개한 일반 OBB 충돌 검출 방법은 상자의 각 꼭지점에 대해 충돌 판정을 해야 하며 양쪽 상자의 모든 꼭지점을 조사하려면, 8×2회의 검사가 필요하다. 또한 각 상자는 6개의 면을 가지므로 총 검사 회수는 6×8×2 = 96회가 된다.

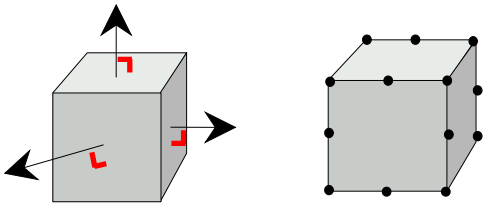

그림 6. 사이점의 추가
Fig. 6. additional middle point

충돌 검사의 정밀도를 높이기 위하여 본 논문에서는 사이점을 추가하는 방법을 사용하였다. 즉 일반 OBB 충돌 검출에서는 꼭지점 8개에 대한 충돌 판정을 하지만 여기에서는 그림 6와 같이 각 꼭짓점 사이에 사이점을 추가하여 충돌의 정밀도를 높이고자 한다.

사이점은 일반 OBB 충돌이 검출하지 못하는 그림 5와 같은 상황을 어느 정도 검출할 수 있게 한다. 또한 사이점의 개수를 증가시키면 정밀도는 더욱 높아진다.

실험을 위하여 본 논문에서는 사이점이 없는 경우를 R, 그리고 사이점의 개수에 따라 T1, T2, T3등으로 명명하였다. 즉 T1은 각 꼭지점 사이에 사이점 1개가 추가된 것이며 결국 총 12개의 사이점이 추가된다. 따라서 두 상자의 충돌을 검출하기 위해 6×(8+12)×2 = 240회의 검사가 필요하다.

하지만 이 검사 회수는 최악의 상황을 가정한 것이며 어느 한 점이라도 충돌로 판정되면 검사는 거기에서 멈추어진다. 즉 일반 OBB 충돌 검출에서 얻은 검사 회수 96회와 사이점 하나를 추가한 T1의 검사 회수 240회는 2.5배이지만 실제로는 일반 OBB 검출이나 T1이 일단 한 지점의 충돌을 검출하면 더 이상의 검사가

필요 없으므로 충돌 검출하는 데는 2.5배의 시간이 걸리지 않는다.

<표 1>에는 각 실험 모델에 따른 최대 검사 회수와 추가 되는 사이점을 나타내었다.

표 1. 모델별 검사 정점과 검사 회수

Table 1. check point and check time in each model

모델	사이점 개수	총 정점수	최대 검사 회수
R	0	8	96
T1	12	20	240
T2	24	32	384
T3	36	44	528
T4	48	56	672
T5	60	68	816
T6	72	80	960
T7	84	92	1104

본 논문에서는 사이점 추가 시 균등 간격으로 배치하였다. 즉 T1은 두 꼭지점의 중간점을 택하였고 T2는 삼등분하는 지점에 두개의 사이점을 위치시켰다.

$$g = \frac{L}{n+1} \tag{3.1}$$

식 (3.1)은 사이점의 간격 g를 알려준다. 여기서 L은 변의 길이이며 n은 사이점의 개수이다.

그림 6은 한 개의 사이점을 추가한 T1을 도식화 한 것이다. 모델 번호가 T1, T2, T3로 커짐에 따라 사이점의 개수가 증가함으로 이를 정밀도가 높아지는 정밀도 단계(level)로 간주할 수 있다. 단 식 (3.1)에서도 볼 수 있듯이 객체의 크기가 커짐에 따라 같은 단계에서도 사이점 간격은 멀어진다.

4. 실험 및 결과 분석

실험을 위하여 기준이 될 완전 충돌 검출 모델(Full Collision Detection Model)이 필요하다. 이것은 연산시간을 고려하지 않고 100%에 가까운 충돌 검출이 가능해야하는 모델이다. 본 실험에서는 상자의 각 변에 모든 점을 검사하는 방법으로 완전 충돌 검출 모델을 만들었다. 매 실험은 각 모델들을 30번씩 수행하여 오차수의 평균치를 결과로 산출했다.

첫 번째 실험에서는 모델에 따른 오차를 보기 위하여 충돌 검사 영역(collision space)의 각 변의 크기를 OBB 변 크기의 5배로 설정하고, OBB 개수(볼륨 수)를 10개부터 45개까지 늘리면서 R과 T1에서 T7, 그리고 완전충돌 모델을 실행했다. 각 모델들의 결과는 연산시간, 충돌횟수, 오차수를 출력하였다.

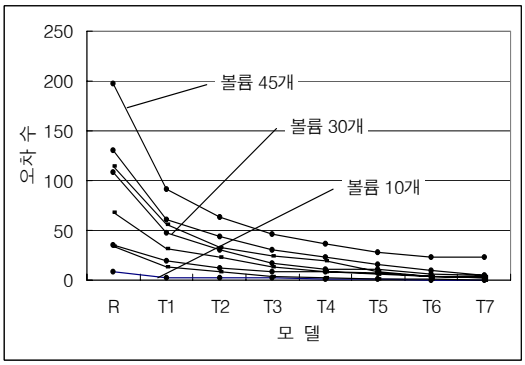

그림 7. 볼륨 수별 모델에 따른 오차수의 변화

Fig. 7. error variety based on each model

그림 7에서 각 볼륨 수에서 모델 번호가 높아갈수록 급격히 오차가 줄어들다가 서서히 감소하는 것을 볼 수 있다. 또한 객체가 많을수록 R과 T1의 오차의 차이가 크다. 이것은 충돌 횟수가 많을수록 오차의 차이가 커진다는 것을 나타낸다. 반대로 객체 수가 적을수록 오차변화율이 적은데, R~T1, T1~T2, T2~T3 구간을 살펴보면 그 차이를 좀 더 자세히 알 수 있다. 객체수가 10개인 경우는 T1에서의 오차 수가 T7과 비슷하므로, T1에서도 정확도 높은 검출(detection)이 가능하다. 반면에 객체수가 45개인 경우는 T3~T5구간의 수치가 T7과 비슷하다.

그림8은 모델 번호를 높일 때마다 오차가 감소하는 비율을 보여준 것이다. 대체적으로 T1에서의 오차 감소율이 가장 컸으며, T5, T6, T7 모델의 오차 감소율은 작았다.

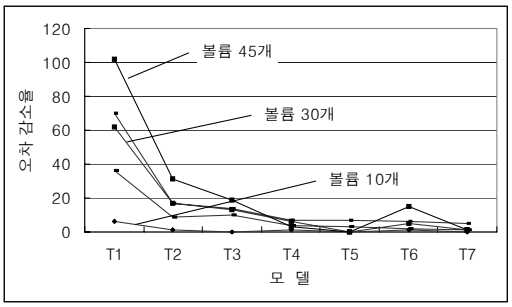

그림 8. 볼륨 수벌 모델에 따른 오차 감소율
Fig. 8. error decrease rate based on each model

그림 9은 모델에 따른 연산시간의 변화를 보여준 것이다. 전체적으로 볼륨수가 많을수록 연산시간이 더 오래 걸리며, 모델에 따라서는 T1에서 T7로 갈수록 시간이 더 걸린다. 그림 7과는 대조적으로 T4~T5, T5~T6, T6~T7 구간의 시간 증가율이 볼륨수가 증가할수록 커지고 있다.

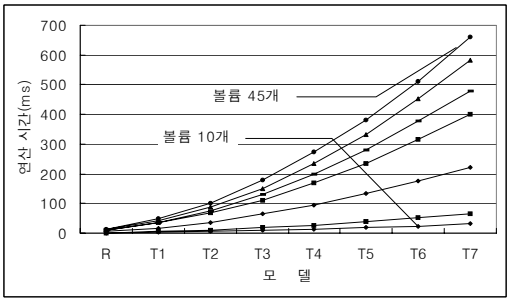

그림 9. 볼륨 수별 모델에 따른 연산시간의 변화
Fig. 9. operating time variety based on each model

두 번째 실험은 OBB 크기(객체 크기)가 오차율과 연산 시간에 영향을 주는 정도를 알아보고자 했다. 충돌 검사 영역(collision space)과 OBB 개수를 고정시키고 OBB 크기에 변화를 주면서 실험했다.

그림 10는 볼륨 수가 20개일 때, OBB의 크기를 증가시키면서 연산시간의 관계를 알아본 것이다. 볼륨수가 많다고 해서 시간이 오래 걸리지 않는다. 즉 연산시간은 OBB 크기와 큰 연관 관계가 없음을 알 수 있었다.

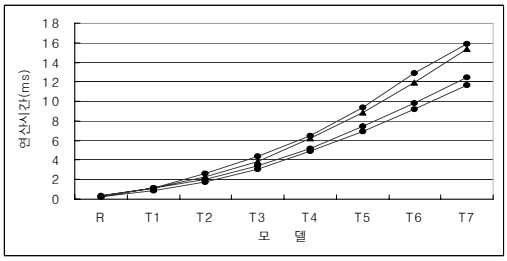

그림 10. 볼륨 크기에 따른 연산시간의 변화
Fig. 10. operating time variety based on volume size

이 실험을 통해 다음과 같은 결론을 얻을 수 있었다.

① 연산 시간은 객체 크기와 무관하다.
② 충돌이 많이 발생하는 상황일수록 연산시간이 비교적 일정하게 증가한다.
③ 볼륨 수가 적을수록 모델의 번호가 커질수록 오차도 작아지지만 오차의 변화가 큰 구간이 있으며 이것은 볼륨 수에 따라 달라진다.

③의 결론은 볼륨 수가 적을수록 충돌 가능 횟수가 적어지기 때문이다. 생성된 볼륨 수가 적을수록 충돌 횟수도 줄어들고 따라서 오차도 줄어들기 때문이다. 결국 충돌 검사 영역의 크기를 충돌이 가장 적게 발생되도록 설정하는 것으로 연산시간을 줄일 수 있다.

이 실험 결과는 3D 객체 충돌에 다음과 같이 이용될 수 있을 것이다. 먼저 실험과 유사한 조건이라면, 충돌 검사 영역안에 몇 개의 객체가 있는지를 조사한다.

예를 들어 객체 45개가 존재하는 3D 장면을 가정해 보자. 그림 8에서, 객체가 45개일 때, T4나 T5 에서 오차감소가 미미함을 알 수 있다. 또한 그림 9에서 보면, 객체가 45개일 때, T3~T5구간의 연산시간의 증가가 빠르게 나타난다. 따라서 T4모델을 선택하면 최적의 선택이 가능하다.

또 다른 경우로 객체 10개가 존재하는 상황을 살펴보자 그림 7에서 볼 수 있듯이 T1 이후의 모든 모델이 비슷한 정도의 오차수를 보인다. 하지만 그림 9의 연산시간은 모델에 따라 지속적인 증가를 보인다. 따라서 T1을 택하여 빠른 연산속도와 최적의 검출 정밀도를 얻을 수 있다.

5. 결론 및 연구방향

본 논문에서는 상자 모양의 3D 객체의 충돌을 검출하기 위하여 점과 면의 충돌 검사에 기반을 둔 OBB 충돌 모델을 살펴보았다. 단순함과 빠른 속도의 장점에도 불구하고 모델의 한계로 인해 검출할 수 없는 충돌상황이 있었다. 이를 보완하기 위해 사이점을 추가하여 검출 정밀도를 높이고자 하였다. 정밀도 증가를 확인하기 위하여 고정 공간 속에서의 사이점 개수에 따른 오차를 비교하였고, 검출 시간을 비교하였다. 나아가 객체크기의 변화에 따른 정밀도 변화를 살펴보았다. 사이점 추가는 정밀도에 영향을 주었지만 5개 이상의 추가는 미미한 정도의 개선을 보였다. 하지만 객체 크기의 변화가 정밀도를 좌우한다는 것은 확인할 수 없었다.

이번 실험에서는 정적 자료만 이용했기 때문에 객체의 이동 속도에 따른 충돌 검출에서 사이점의 효과를 확인하지 못했다. 따라서 앞으로 이동 객체 충돌 검출에서의 사이점의 역할을 밝혀나가고자 한다.

시간과 정확도를 모두 만족시킬 수 있는 엔진을 개발하는 것은 시간과 자원의 소모가 큰 작업이다. 본 논문에서 제시한 사이점 추가에 따른 정확도의 변화와 충돌 검출 시간의 변화는 이러한 복잡한 작업에 기준을 제시하여 줄 것이다.

참 고 문 헌

[1] Nick Bobic, "Advanced Collision Detection Techniques," http://gamasutra,com/features/20000330.bobic_-1.htm, 2000

[2] David Burg, Physics for Game Developers, O□□ Reilly & Associates, Inc., 1999

[3] David Elberly, "Dynamic Collision Detection Oriented Bounding Boxes," http://www.magic-software.com., 2002

[4] Daved Eberly, "Intersection of Convex Object : The Method of Separating Axes," http://www.magic-software.com, 2001

[5] Mark Deloura, Game programming Gems, CHARLES RIVER MEDIA, INC., 2001

[6] Eric Lengyel, Mathematic for 3D Game Programming Computer Graphics, CHARLES RIVER MEDIA, INC., 2002

[7] 김병기, “Collision Detection,” 민프레스 세미나, 2002

[8] 이만희, “Fast Overlap Test for OBB,” http://user.collian.net/~manilee/Fast_Overlap_Test_for_OBBs.pdf, 2002

[9] 이영재. “ 컴퓨터 게임을 위한 2D 충돌 감지 알고리즘 비교 분석에 관한 연구,”한국게임 학회 논문지 제1권, 제1호, pp.42-48, 2001

[10] http://www.gpgstudy.com

[11] http://www.magicsoftware.com

[12] http://www.netian.com/~proline1

[13] http://www.wzsoft.com/~leechen

저 자 소 개

姜 允 美 (學生會員)

2002년 단국대학교 전자계산학과 (학사)
2002년~ 현재 단국대학교 전자계산학과(석사과정)
관심분야 : 컴퓨터 게임, 리팩토링

朴 容 範 (正會員)

1985. 서강대학교(학사)
1987. N.Y. Polytechnic Univ.(석사)
1991. N.Y. Polytechnic Univ.(박사)
(현) 단국대학교 전자계산학과 교수
관심분야 : 컴퓨터 게임, 패턴인식, 인포메이션 아키텍처