韓國 컴퓨터情報學會 論文誌
第11卷 第3號, 2006. 7.

2006-11-3-1-22

사각형 충돌감지알고리즘을 사용한 슈팅게임 구현

서 정 만*, 한 상 훈**, 이 호**

Implementation of Shooting game using collision detection algorithm of

Jeong-Man Seo*, Sang-Hoon Han**, Ho Lee**

요 약

PC환경에서 슈팅게임 시에 충돌이벤트에 대한 내용과 충돌감지 알고리즘에 대하여 소개하였고, 기존의 사각형 충돌감지 알고리즘 기법에서 단순한 사각형 충돌의 단점을 보완한, 작은 사각형 단위의 충돌 체크 기법을 제안하여 화면디자인과 함께 단순한 슈팅게임을 구현하였다. 실험과 실제 구현한 게임 화면 디자인을 통하여 제안한 알고리즘이 실제 게임에서 적용할 수 있음과 기존의 알고리즘보다 제안한 알고리즘이 우수함을 보였다.

Abstract

We reviewed the collision event with pc-based shooting games and existing collision detection algorithms. Then we proposed a new collision check technique using a small quadrilateral unit, by which existing quadrilateral collision detection techniques can be made up for. For demonstration we implemented a simple shooting game having its screen output. We proved that the proposed technique can be applied to real computer games by means of showing the experimental results and screen outputs from implemented real games.

▸ Keyword : collision technique, collision algorithm, collision detection

---

• 제1저자 : 서정만
• 접수일 : 2006.06.01, 심사일 : 2006.06.27, 심사완료일 : 2006.07.18
* 국립한국재활복지대학 컴퓨터게임개발과 교수, ** 국립한국재활복지대학 컴퓨터정보보안과 교수

Ⅰ. 서 론

게임이라는 용어는 “흥겨웁게 뛰다”라는 인도 유러피안 계통의 “ghem”에서 파생된 단어로 ”흥겹다“는 정신적으로 재미 또는 즐거움을 느낀다는 뜻과 ”뛰다“라는 동작을 나타내는 동사적 의미의 단어가 합성된 용어이다[1]. 문화콘텐츠의 중요성이 증대되면서 영상, 음반, 애니메이션, 게임 등 문화콘텐츠산업은 전 세계적으로 급속하게 성장하고 있다. PC게임, 비디오게임, 업소용 게임, 온라인게임, 모바일게임 등 게임산업의 각 장르도 그 성장률은 계속 높아지고 있다. 특히 첨단 제작기술 및 이동통신 기술의 발전에 따라 게임이 구현되는 플랫폼도 기존 오프라인과 온라인으로부터 모바일에 이르기까지 다양해지고, 이에 따라 게임제작도 더욱 첨단화되고 있다[2]. 인터넷을 이용한 온라인게임이나 온라인 기능을 탑재한 비디오게임기의 등장 및 모바일게임의 보급은 게임이라는 문화콘텐츠의 접근이용도를 높이고 이는 결국 앞으로 보다 거대한 게임시장의 형성으로 이어지게 될 것이다. 또한 우리나라 모바일기기의 보급률과 온라인게임의 선전을 고려했을 때, 모바일게임에 대한 투자와 지원은 게임산업을 통한 국내 산업의 견인과 모바일게임의 해외수출 선점 효과로 이어질 것이다. 오락실에서 이루어지고 있는 게임들은 대부분 아케이드 게임의 형태에 속한다고 볼 수 있다. 인터넷을 이용하지 않은 PC에서 이루어지고 있는 슈팅게임은 형태가 여러 가지이지만, 대부분 총알이나 어떤 형태의 무기를 발사하고, 그것이 적기나 아군이 맞으면 충돌이 되어서 폭파되거나 없어지도록 하는 것이다. 여기에 슈팅게임에서 충돌이 되었는지를 검사하거나 프로그램하는 것은 쉬운 일은 아니다. 본 논문에서는 기존의 사각형을 이용한 충돌처리 기법 알고리즘에 대하여 알아보고, 문제점을 제시하며, 단순한 사각형 충돌의 단점을 보완한, 작은 사각형 단위의 충돌 체크 기법을 제안하고, 실험과 실제 구현한 게임 화면 디자인을 통하여 제안한 알고리즘의 우수성과 향후 연구과제에 대하여 논한다.

Ⅱ. 기존 연구

충돌 처리란, 말 그대로 두 물체가 서로 충돌 관계에 있는지를 판별하는 작업을 의미 한다. 이러한 처리가 필요한 이유는, 디자인과 프로그램을 멋지게 구현하였더라도 화면에 등장하는 아군 캐릭터 그리고 화면 상단에 존재하는 적군 캐릭터들, 멋진 등장 뒤에 멋있게 총알 발사 버튼을 누르고, 그리고 그 화면 상단으로 날아가는 총알을 바라보며 그 총알에 맞아 죽게 될 적을 상상하지만, 만약 충돌처리가 이루어지지 않는다면 아무리 총알을 쏘고 그 총알이 적들에게 날라가도 그 총알에 맞아 죽지 않는 적들을 본다면 아마도 허무감이나 허탈감, 아니 이런 게임이 다 있는가 하고 생각할 것이다. 컴퓨터 게임의 경우 충돌은 가상과 현실과의 상호작용을 제공하고, 인공지능과 더불어 게임을 재미있게 만드는 중요한 요소이다. 그러나 게임의 규모가 방대해지면서 인공지능은 예전에 사용했던 기술 이상으로 인공지능 기술이 필요하게 되었고[3], 충돌에 대한 종류를 규정짓는 것이 필요하다. 슈팅(Shooting)게임에서의 충돌 이벤트는 다음과 같이 분류할 수 있다.[4]

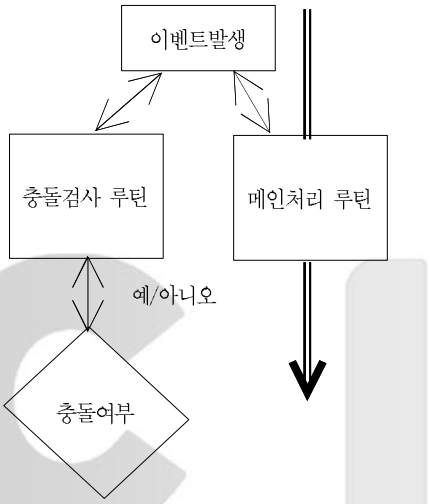

그림 1. 충돌검사 루틴
Fig 1. routine of collision check

* 충돌 이벤트

① 적 무기에서 아군 캐릭터와의 충돌 여부를 감지
② 아군 캐릭터에서 적 무기와의 충돌 여부를 감지
③ 아군 무기에서 적 캐릭터와의 충돌 여부를 감지
④ 적 캐릭터에서 아군 무기와의 충돌 여부를 감지
⑤ 아군 캐릭터에서 적 캐릭터와의 충돌 여부를 감지
⑥ 적 캐릭터에서 아군 캐릭터와의 충돌 여부를 감지

위와 같이 아군 캐릭터와 적군 캐릭터 사이의 충돌, 적의 무기, 총알, 획득해야 할 아이템과의 충돌, 장애물과의 충돌까지 충돌체크 루틴이 필요하다. 이를 루틴으로 표현해 보면 그림1과 같다[5]. 다음은 충돌검사 함수에 대한 알고리즘을 나타내고 있다.

```
int 충돌검사_루틴()
  {
    int col_flag = 0;
    충돌 감지 알고리즘 수행;
    if(충돌감지) col_flag = 1;
    return col_flag;
  }
```

III. 충돌 감지 알고리즘

충돌 처리에는 여러 가지 기법들이 존재한다. 여러 가지 충돌 처리에 대해서 소개하면, 원을 사용하는 방법, 2D에서 가장 정확한 판단을 할 수 있는 픽셀 처리 방법, 이미지의 전체 영역을 작은 박스 단위로 나뉘어서 비교적 정확한 판단을 할 수 있는 방법, 회전 사각형 충돌 처리, 그리고 가장 빠르고 쉽게 구현할 수 있는 사각형 충돌 기법들이 있습니다. 그 외에 3D에서 사용할 수 있는 바운딩 박스에 의한 충돌 기법, 바운딩 구에 의한 충돌 기법들이 있다. 충돌 처리에 이 처럼 여러 가지 기법이 존재하는 이유는, 보다 정확한 충돌 처리를 하기 위함과 정확하지 않지만, 특정 게임의 생명이라 할 수 있는 속도 문제들로 인해 이렇게 여러 다양한 기법들이 존재하게 된다[6]. 격투 게임과 같이 비교적 많은 객체들이 존재하지 않고 정확히 특정 부분에 충돌이 발생 했는지를 알아보려면, 속도는 느리지만 픽셀 단위 혹은 작은 사각형단위의 충돌 처리가 적합할 것 이고, 빠른 그래픽 처리 속도를 요구하는 슈팅게임 등에서는 비교적 정확하진 않지만 코드가 단순해서 빠른 처리를 할 수 있는 사각형 충돌 처리가 적당할 것이다[6]. 물론 이 두 가지 장단점을 보완한 충돌처리 기법도 존재한다. LSP 충돌처리라는 기법이 있는데, 이는 라인 단위로 충돌 처리를 하는 기법으로 픽셀 처리 보다 빠르고 사각형 충돌 처리만큼이나 속도도 나타내는 기법이다.

3.1 원을 사용한 방법

원을 사용하는 방법에서는 이들의 반지름을 사용한다. 즉 이들 두 원의 반지름 합을 상수로 정해 이 상수보다 같거나 작은 경우 충돌로 감지하는 방법이다. 그림2는 원을 사용한 충돌감지에 대한 것을 나타내고 있다.

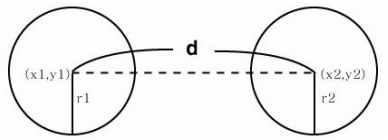

그림 2. 원을 사용한 충돌 감지
Fig. 2. Collision detection using circle

두 원 사이의 거리 d는 식(1)과 같이 나타낼 수 있다.

$$d = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2} \quad \cdots\cdots (1)$$

d 두 원간의 거리

이 경우 $d \leq r_1+r_2$ 이라면 두 원의 충돌이 발생한 경우이다. 원 둘레에 대한 좌표 값은 $x_n^2 + y_n^2 = r_n^2$ 를 사용해 구할 수 있다.

원 사용방법은 접촉, 충돌 감지를 위한 계산이 간단한 장점을 가지고 있으나 원 형태가 아닌 경우 예를 들면 직사각형의 형태를 가지고 있는 경우엔 적합하지 않으므로 제한적으로 사용되고 있다.

3.2 사각형을 사용한 방법

그림3은 사각형을 사용한 충돌감지에 대한 것을 보여주고 있다. 사각형은 4개의 특징점 좌표를 가지고 있으며 이들은 동일한 x 좌표, y 좌표를 가지고 있으므로 이들의 좌표값을 이용하여 충돌여부를 확인할 수 있다[7].

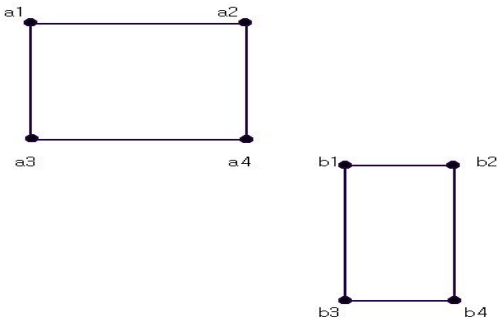

그림 3. 사각형을 사용한 충돌 감지
Fig. 3. Collision detection using rectangular

예를 들면 a4의 좌표를 (x4,y4)라 하고 b1의 좌표를 (x1′,y1′)라 하면 (x1′>x4 ‖ y1′> y4) 경우와, a1의 좌

표를 (x1,y1)라 하고 b4의 좌표를 (x4′,y4′)라 할 때 (x1>x4' || y1 > y4') 경우 중 하나만 만족하면 충돌이 발생하지 않은 경우이다. 이 방법은 충돌체크가 간단한 장점은 있으나 오브젝트의 형상(예: 삼각형)에 따라 정밀한 감지가 어렵기 때문에 환경에 따라 제한적으로 사용할 수 있다.

3.3 제안한 사각형을 사용한 방법

앞에서 언급한 충돌처리를 요약하면 3가지 경우로 볼 수 있는데, 충돌상황이 발생하는 경우는 첫째로, 아군 무기에서 총알 발사 시 적 캐릭터와 충돌, 둘째로, 적 캐릭터의 무기 발사 시 아군캐릭터와의 충돌, 셋째로, 아군 캐릭터와 적 캐릭터와의 충돌이다.

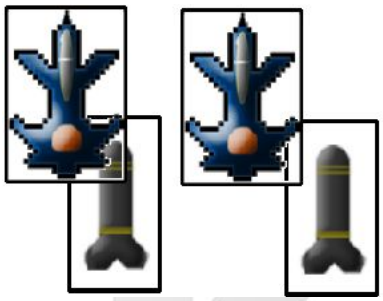

그림 4. a) 충돌 캐릭터, b) 충돌되지 않은 캐릭터
Fig 4. a) collided character b) non-collided character

현재 게임에서 사용하는 사각형 충돌에 대해서 알아보면, 사각형 충돌은 상당히 단순한 구조를 가지고 있다. 또한 쉽게 구현이 가능한 만큼 여러 부분에서 사용이 가능하며, 그림 4에서 보면 사각형 충돌이 어떠한 것이라는 것을 알 수가 있다. 왼쪽의 그림 (a)를 보면, 정확히 충돌이 되었다는 것을 알 수 있다. 그러나, 오른쪽의 그림 (b)를 보게 되면, 이미지들의 주위에 둘러져 있는 사각형은 충돌이 되었지만, 실제 이미지들(2개의 캐릭터)는 상호간에 충돌되지 않았음을 확인할 수 있다. 그림 4의 (b)처럼 충돌되지 않았음에도 불구하고, 충돌된 것으로 처리됨으로 인하여 게임의 신뢰성이나 게이머로부터 불만을 가지게 될 수 있다. 이 처럼 사각형 충돌은 픽셀 단위 비교가 아닌 이미지가 가지고 있는 좌표값과 이미지의 X, Y 크기를 가지고 만든 사각형으로 두개의 이미지가 충돌이 발생 했는지 판단하는 것으로 오른쪽 그림과 같은 문제점을 발생하게 된다. 이러한 문제에 대하여 단순한 사각형 충돌의 단점을 보완한, 작은 사각형 단위의 충돌 체크 기법을 제안하였다. 큰 이미지의 영역을 작은 사각형 단위로 잘라 이 잘린 영역 별로 충돌을 비교하는 이 비교 방법은 위의 사각형 충돌 시 발생했던 문제점을 해결할 수 있다. 알고리즘을 식으로 표현하면 다음과 같다.

만약 캐릭터 2개를 각각 A, B라고 가정하면

```
If (A.Left >= B.Left) And _
        (A.Left <= B.Left + A.Width) ····· (2)

If (B.Top >= A.Top) And _
        (A.Top <= B.Top + B.Height) Then ····· (3)
```

표 1. 캐릭터 속성
Table 1. character Attribute

속 성	의 미 설 명
Hight	컨트롤의 크기, 틀 전체의 높이를 의미.
Width	컨트롤의 전체 넓이를 의미.
Top	컨트롤의 상단 좌표를 의미, 폼의 최 상단은 0.
Left	컨트롤의 왼쪽 좌표를 의미, 폼의 최 좌측 좌표는 0.

알고리즘에서 식(2)와 식(3)을 동시에 만족하면 2개의 A,B 캐릭터가 충돌된 것이다. 식에서 Top, Width, Height등을 점차로 잘게 쪼개 반복 처리한다면 실제적으로 충돌처리 여부를 세밀한 부분까지 처리할 수 있다. 여기서 각각의 설명은 표 1에 나타나 있다.

Ⅳ. 실험 및 구현

4.1 화면 디자인

그림 5는 게임구현 시 화면디자인을 보여주고 있다[8]. 표 5, 표 6에 개체의 속성 값을 보여주고 있다. Form 1개, Picture Box 5개, Timer 4개, MMC 컨트롤 3개, Command Button 2개, Label 2개, Text Box 2개로 구성되어 있다.

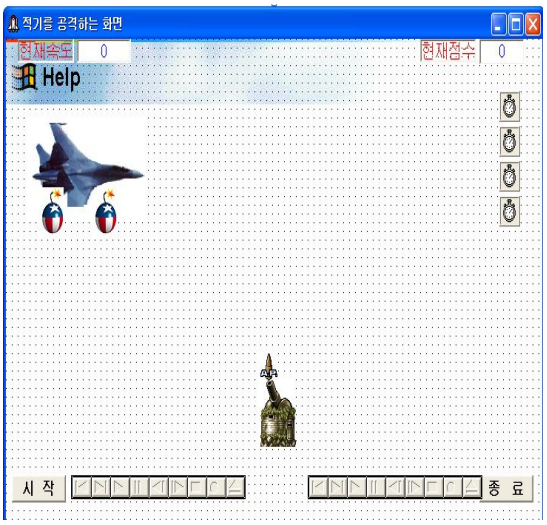

그림 5. 화면 디자인
Fig 5. window design

표 5. 개체 속성값1
Table 5. value 1 of object attribute

개 체	속 성	값
Picture Box-1	Name Picture Autoredraw Borderstyle	Pic_Plane "적기-1" True 0-없음
Picture Box-2	Name Picture Autosize Borderstyle	Pic_attacker "공격대-1" True 0-없음
Picture Box-3	Name Picture Autosize Borderstyle	Pic_missel "미사일" True 0-없음
Picture Box-4	Name Picture Autosize Borderstyle	pic_star1 "미사일-1" True 0-없음
Picture Box-5	Name Picture Autosize Borderstyle	pic_star2 "미사일-2" True 0-없음
Form	Caption Backcolor Name	적기를 공격하는 화면 흰색 Form_Main

MMC 컨트롤은 배경음악과 미사일을 발사 할 때의 소리와 충돌이 되었을 때의 소리를 위한 개체이다.

표 6. 개체 속성값2
Table 6. value 2 of object attribute

MMC-1	Name	Mci_First
MMC-2	Name	Mci_Second
Command Button-1	Name Caption	Cmd_start 시작
Command Button-2	Name Caption	Cmd_stop 종료
Label-1	Name Caption	Label_speed 속도
Label-2	Name Caption	Label_jumsu 점수
Timer-1	Name Enabled Interval	Timer_plane False 50
Timer-2	Name Enabled Interval	Timer_missel False 100
Timer-3	Name Enabled Interval	Timer_bkmusic False 15000
Timer-4	Name Enabled Interval	Timer_star False 50
TextBox-1	Name Text	Text_speed 0
TextBox-2	Name Text	Text_Jumsu 0

4.2 실험환경 및 결과

실험환경으로는 프로그램 언어는 visual basic을 사용하였고, 펜티엄4, 윈도우 XP, 메인메모리 512메가바이트이다. 실험결과는 기존의 사각형 충돌알고리즘은 성공률이 81%이지만, 제안한 알고리즘은 97%의 성공률을 보이고 있다. 실제 구현한 화면에서 정상적으로 충돌처리 됨을 알 수 있었고, 기존의 사각형 알고리즘에 비하여 개선된 제안한 알고리즘의 성공률이 높아졌음을 알 수 있었다.

표 7. 실험결과값
Table 7. Values of experimental result

알고리즘	시도횟수	성공	실패	성공율
기존알고리즘	100	81	190	81%
제안한알고리즘	100	97	3	97%

V. 결론

본 논문에서는 PC환경에서 슈팅게임 시에 충돌이벤트에 대한 내용과 충돌감지 알고리즘에 대하여 소개하였고, 기존의 사각형 충돌감지 알고리즘 기법에서 단순한 사각형 충돌의 단점을 보완한, 작은 사각형 단위의 충돌 체크 기법을 제안하여 화면디자인과 함께 단순한 슈팅게임을 구현하였다. 슈팅게임 구현에서 큰 이미지의 영역을 작은 사각형 단위로 잘라 이 잘린 영역 별로 충돌을 비교하는 이 비교 방법은 본 논문에서의 사각형 충돌 시 발생했던 문제점을 해결할 수 있음을 보였다. 향후 연구과제로는 빠른 슈팅게임에서 제안한 사각형 충돌감지 알고리즘이 속도를 감안하여 어느 정도까지 세밀하게 충돌을 감지하는 것에 대하여 연구하여야 할 것이다.

참고문헌

[1] 민용식, 이동희, “게임학개론”, 도서출판정일, pp. 14, 194, 2002

[2] 성재환, "모바일 게임 산업 동향과 발전방안연구“,(재)게임종합지원센터 , pp.11, 2001

[3] 이세일, “게임 적용을 위한 Dynamic Programming 알고리즘 길찾기”, 한국컴퓨터정보학회 논문지, 제10권 4호, pp214, 2005

[4] 홍석기 외2, 게임 하듯이 게임 만들기,pp304-309, pp416-420, 가남사, 1998

[5] 이영재, “게임을 위한 2D 충돌감지알고리즘 비교분석에 관한 연구”, 한국게임학회 논문지, 제1권 1호, pp.43-45, 2001

[6] http://www.cizi.pe.kr/Lec/Invader19.htm, "충돌처리“

[7] 주정규, “격투게임 제작기법과 알고리즘에 관한 연구”, 한국게임학회 논문지, 제2권 1호, pp71-74, 2002

[8] 서정만, “비주얼베이직과 게임만들기”, 도서출판정일, pp. 319-322, 2005

저 자 소 개

서 정 만

2003년 충북대학교 컴퓨터공학과 공학박사
1988년~1993년 엘지전자 컴퓨터연구소 주임연구원
1993년~2000년 삼성중공업 중앙연구소 선임연구원
2000년 ~ 2002년 : 극동정보대학 컴퓨터게임개발과 교수
2002년 ~ 현재 : 국립한국재활복지대학 컴퓨터게임개발과 조교수
<관심분야> 데이터베이스, 게임프로그래밍, 실시간처리

한 상 훈

1995년 동국대학교 대학원 컴퓨터공학과 석사
2002년 동국대학교 대학원 컴퓨터공학과 공학박사
2003년 ~ 현재 : 국립한국재활복지대학 컴퓨터정보보안과 조교수
<관심분야> 정보보안, 형태인식, 컴퓨터 비전, 멀티미디어

이 호

1989년 벨기에 VUB 대학원 정보공학과 석사
2002년 성균관대학교 대학원 정보공학과 공학박사
1982년 ~ 1991년 한국전자통신연구원 선임 연구원
2002년 ~ 현재 : 국립한국재활복지대학 컴퓨터정보보안과 부교수