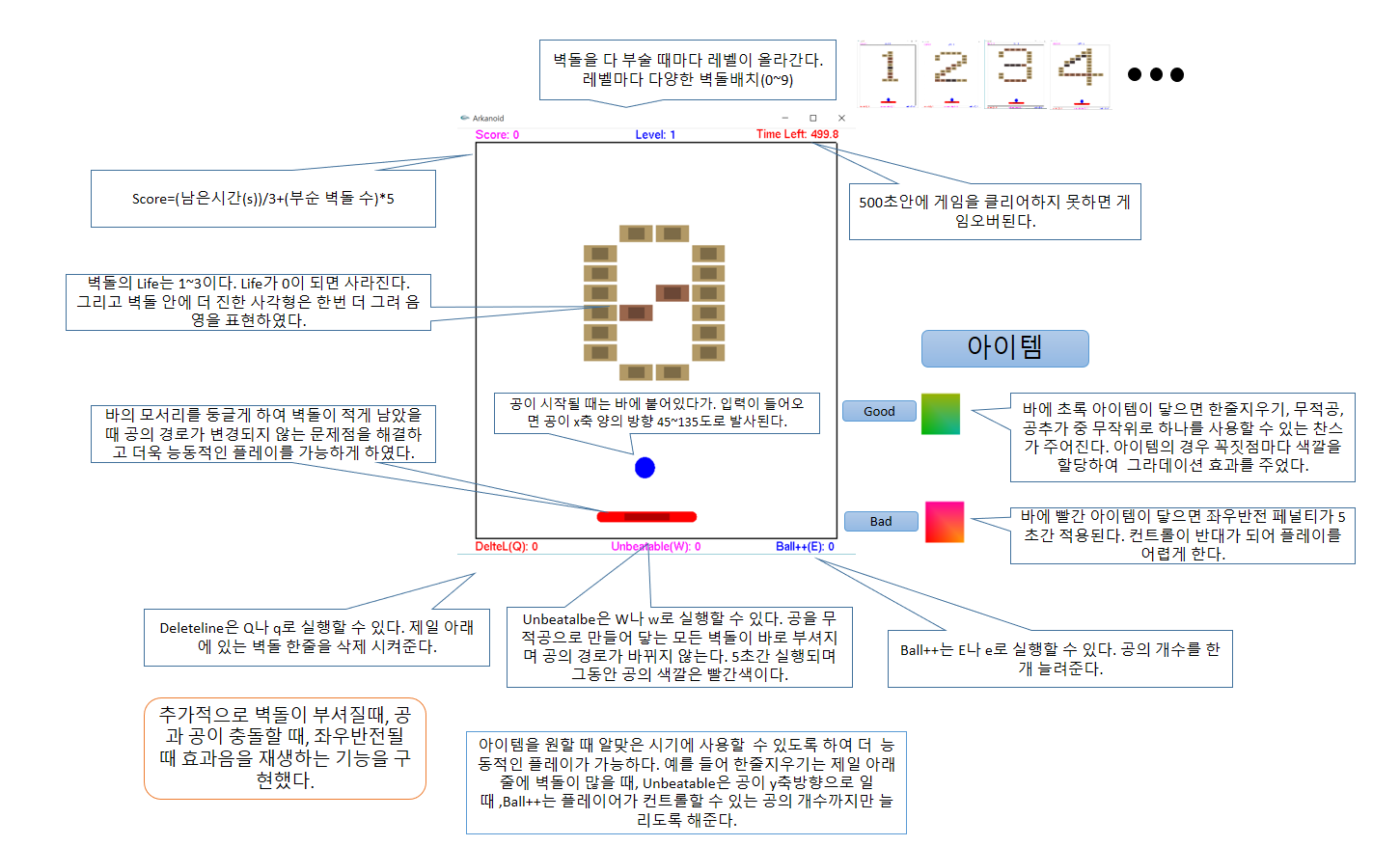
**프로그래밍 방법론 10조 프로젝트 보고서**

전기정보공학부

2015-12309 강재신

2015-14392 김찬우

2015-13741 오경록



Code Compile and Test Environment

OS: Windows10

Compiler: Microsoft Visual Studio 2015 Community

CPU: Intel i5-6200U

RAM: 8GB

GPU: Intel HD Graphics 520

Visual Studio의 컴파일 결과 실행파일 exe가 생성되기 때문에 Windows 환경에서만 실행된다. 만약에 Linux 같은 다른 OS에서 실행하려면 GCC같이 실행 환경에 맞는 컴파일러를 사용해야한다. 그리고 결정적으로 효과음을 재생할 때 Windows.h을 Include하고 WinAPI함수인 PLAYSOUND를 사용하였기 때문에 이 코드를 그대로 다른 OS에서 컴파일해 실행하는 것은 불가능하다. 그리고 exe파일과 glut.32lib, \*.wav 파일이 같은 디렉토리안에 있어야 한다.

Abstract

벽돌 깨기(Breakout game)는 1970년대에 만들어지고, 현재까지도 많은 컨텐츠를 추가하면서 인기 아케이드 게임의 자리를 유지하고 있다. 벽돌 깨기 게임은 화면 하단의 바(bar)를 움직여 공을 튕겨내어 화면 상단에 위치한 벽돌을 모두 부수는 간단한 게임 규칙을 가지고 있다. 그렇기 때문에 게임 중에서는 구현하기 쉬운 편에 속한다. 이번 프로젝트에서는 기본적인 벽돌 깨기 게임을 구현하고 아이템을 추가하여 Arkanoid 버전까지 재현하려고 하였다. 우리가 구현한 게임은 총 10개의 stage가 존재하며, 획득 시 화면이 좌우로 반전되는 자동효과 아이템과 획득 시 다양한 효과가 있는 사용 아이템을 가지고 있는 것이 특징이다. 추가로 게임에서 감초 역할을 하는 효과음을 집어넣어 게임이 좀더 재미를 느낄 수 있게 하려고 한다.

Variable Introduction

● #define 변수

gridrow와 gridcol은 배열된 벽돌 array의 열과 행의 크기이다. 그래서 벽돌을 초기화할 때 grid상의 좌표로 위치를 설정할 수 있도록 하였다. boundaryleft, boundaryright, boundaryup, boundarydown으로 공이 움직이는 경계 범위를 설정하고 interval로 벽돌 사이의 간격(벽돌이 구분되도록)을 만들었다. FPS로 시간 단위를 바꾸고 WindowWidth, WindowHeight로 윈도우의 크기를 설정하였다.

● 선언된 변수

함수 내에 선언된 지역변수까지 고려하면 너무 많기 때문에 코드 위쪽에 선언된 변수만 설명한다. Score, level은 각각 게임에서 얻은 점수와 스테이지의 레벨이다. Initialcount는 후 Initshape()함수에서 brick vector의 크기로 초기화된다. Rotation과 unbeatablestate는 bool 타입의 변수로, 처음에 거짓으로 초기화되어있다가 좌우가 반전되는 아이템과 무적 공 아이템을 통해 참으로 바뀌어 아이템 함수가 실행된다. Unsigned int로 선언된 item으로 시작되는 변수는 각 아이템의 소지수를 나타낸다. Deletebricknum은 벽돌이 없어질 때마다 증가하여 점수 계산에 영향을 준다. Bool 타입의 gamestart 변수는 처음에 거짓으로 초기화되어있는데 게임이 시작하면 true로 바꿔 시간을 움직이기 시작하고 공과 바, 아이템의 움직임을 구현한다.

Function Introduction

함수는 크게 다음과 같이 나눌 수 있다.

<GL 기초 함수, 충돌 함수, 아이템 함수, 스테이지 함수, 조작 함수, 텍스트 표시 함수>

이 중에서 GL 함수를 사용하여 창을 띄우고 움직이는 과정을 반복하는 GL 기초함수를 제외한, 독자적으로 작성한 나머지 네 가지 함수를 설명하고자 한다.

● 충돌 함수

결국 모든 충돌은 공의 움직임에 따라 일어나므로, 공을 기준으로 설명하는 것이 좋다. 이 충돌함수는 void idle()에 구현되어있다.

우선 게임이 시작되면 ms 단위까지 시간을 재기 시작한다. 그 후 공의 객체 cir를 cir의 vector size까지 각도를 랜덤으로 설정하여 속도를 부여한다. 처음 시작할 때 공은 1개 만들어지므로 cir,size()=1로 시작할 것이다. 그 후 idle()이 실행될 때마다 공과 아이템을 움직인다.

공과 공 사이의 충돌을 가능하게 하여 게임에서 변수를 만들고자 하였다. 공이 충돌을 감지하면 충돌 공식을 이용하여 공의 속도와 위치를 재설정한다. 공과 공이 실제로 맞았는지는 공이 평면과 부딪히는 경우와 다르게 쉽게 확인하기가 어렵기 때문에 확실히 알기 위해서 “bounce.wav” 효과음을 삽입했다.

공은 움직이면서 벽돌 혹은 게임 속 경계에 부딪히게 된다. 공과 벽돌 사이의 충돌은 변수 unbeatablestate의 참ᆞ거짓 여부에 따라 달라져야 할 것이다. Unbeatablestate가 거짓을 반환할 경우 충돌 당시 공이 벽돌과 이루는 기울기 벡터를 저장하여 물체 충돌 과정을 고친 후 기울기 벡터를 맞은 면에 따라 대칭 변환하여 튕겨나가도록 표현하고, 벽돌의 life가 0이라면 벽돌이 사라지면서 deletebricknum이라는 변수에 1을 더하고 “bomb.wav” 효과음이 나오도록 한다. 벽돌이 사라지면 아이템이 rand() 함수를 통해 설정한 확률로 아이템을 떨어트린다. 만약 Unbeatablestate가 참이라면 튕겨나가는 대신 벽돌과 접촉할 경우 벽돌이 life에 관계없이 바로 사라지도록 하였다.

공이 경계에 맞을 때는 벽돌처럼 어느 면에 충돌할지 고민할 필요 없이, 경계를 구성하는 사각형이 공과 충돌할 수 있는 면에서의 충돌만 고려해주면 된다. 역시 기울기를 저장한 후 면에 따라서 대칭 변환하여 튕기는 것처럼 만들었다.

공이 벽돌이 없는 화면 아래 부분까지 내려온다면 공을 바로 튕겨 다시 위로 올려야 한다. 하지만 사각형으로 바를 만들 경우 충돌 판정을 설정할 면이 평면밖에 없기 때문에 능동적으로 각도 변환이 불가능하며, 게임이 지나치게 단조로워질 수 있다. 따라서 바의 양 끝에 원 모양을 추가하여 능동적으로 공의 방향을 바꿀 수 있도록 만들었다. 충돌 함수 구현 방식은 크게 달라지지 않는다. 주어진 코드에서는 바가 얇아 바의 원 부분에 닿게 하기 어려울 수 있기 때문에 변형하길 바란다.

● 아이템 함수 – void funcitemdelete(), void funcitembeatable(), void funcitemmultiple()

공과 부딪힌 벽돌의 life가 0이 되면 확률적으로 아이템을 떨어뜨린다. 이 확률을 코드에서는 34%로 설정하였다. 34%의 확률을 다시 나눠 절반의 확률로 자동효과로 화면을 좌우로 반전시키는 아이템을, 각각 1/6의 출현 확률로 한 줄을 없애는 아이템, 충돌판정과 life에 관계없이 지나가면서 모든 벽돌을 부수는 공을 만드는 아이템, 시작지점에 공을 새로 만들어 공이 여러 개가 되도록 하는 아이템을 생성한다.

좌우 반전 아이템은 획득 시 rotation 변수를 참으로 바꾼다. Rotation이 참일 때, void renderScene()에서 glRotatef(180,0,1,0) 함수를 이용해 화면을 y축 대칭 시킨다. 이 때 if문을 추가하여 좌우가 반전되었을 때 텍스트의 위치를 재설정하였다. 실제로 아이템 효과가 발동될 경우 마우스와 키보드 입력이 반대로 되는 것처럼 느껴진다. 획득하면 “warning.wav” 효과음이 켜지면서 긴박한 분위기를 조성한다.

Funcitemdelete()에 구현되어있는 한 줄 지우기 아이템은 획득 시 for문을 돌리고 안의 벽돌 vector의 erase함수를 이용하여 가장 밑의 줄의 벽돌이 들어가있는 벽돌 vector의 내용을 지운다. Funcitembeatable()은 벽돌과의 충돌 판정을 무시하는 무적 공으로 바꾸는 함수로, 충돌 함수의 unbeatablestate 변수를 참으로 바꾼다. 좌우 반전 아이템과 무적 공 아이템은 지속시간이 5초로 설정하였고, 무적 공 아이템의 경우 효과가 지속됨을 확실히 보여주기 위해서 공의 색깔을 빨간 색으로 바꾸도록 하였다. Funcitemmultiple()은 새로운 스테이지 시작 시 공을 새로 만드는 것처럼 공을 새로 추가하는 함수이다. 공 추가 아이템 획득 시 공이 새로 추가된다. 바를 움직이면 공이 x축 기준 45°~ 135°의 각도를 랜덤으로 설정해 움직이기 시작한다. 3종류의 아이템 획득 시 “item.wav” 효과음이 켜진다. 일반적인 Arkanoid 게임과 달리 아이템을 원하는 타이밍에 사용할 수 있도록 하여 플레이어가 바 조작 이상으로 능동적으로 게임에 참여하고 효과적으로 벽돌을 부술 수 있도록 하였다.

● 스테이지 함수 – gamedata,h,

스테이지의 구현은 gamedata.h에 되어있다. 스테이지는 맵을 구성하는 벽돌의 life를 배열한 2차원 array로 되어있다. 각 벽돌의 life를 level 함수에 입력하면 해당 life를 가진 벽돌이 위치에 맞게 배열되어 나타난다. 공이 벽돌을 계속 부숴 마지막 벽돌과 충돌 시, idle() 내부의 if문이 활성화되어 renderScene()함수에 switch문으로 구성된 스테이지를 불러오는 역할을 한다. 마지막 벽돌이 사라지면 level이라는 변수가 1이 더해지고, level을 parameter로 받아 switch문의 case가 변화하면서 다음 스테이지가 선택된다. Level은 처음에 1으로 초기화 되어있는데, 마지막 10 스테이지를 클리어 할 경우 default에서 if문에 걸려서 게임이 종료되도록 하였다.

● 조작함수 – void SpecialKey(int key, int x, int y), void MyKeyboardFunc(unsigned char Key, int x, int y), void processMouse(int x, int y)

게임은 플레이어의 능동적인 접근이 필요하다. 플레이어가 게임에 참여하기 위해서 필요한 것이 키보드 조작함수랑 마우스 조작함수이다. SpecialKey함수에서 키보드는 key parameter를 통해 방향키 입력을 받은 후 GLUT\_(방향)\_BUTTON에 따라 switch 문을 나눠 bar가 설정한 velocity를 갖도록 하였다. 좌우조작만 입력받으며 그 외의 방향키 조작은 바의 움직임에 영향을 주지 않는다. 그러나 실제로 실행해보면 키보드로 바를 조작하면 그 속도가 느려서 공을 놓치기 십상이다. 그래서 마우스로 조작하는 함수가 필요하다.

이를 가능하게 하는 함수가 processMouse 함수이다. 받는 argument가 x,y 좌표로 2개이지만 우리가 구현한 코드에는 x좌표에 따른 움직임만 설정하여 마우스에 의해 공의 위치 및 바의 위치가 가려지는 문제를 피하고자 하였다.

아이템을 사용하기 위해 아이템의 단축키를 받는 함수는 MyKeyboardFunc이다. Q(q), W(w), E(e)를 입력하면 key로 받아 switch문을 통해 각각 아이템을 뜻하는 변수가 0 이상일 때 한 줄 지우기, 무적 공, 공 늘리기 함수를 실행하고 변수를 1 줄이도록 하였다.

● 텍스트 표시 함수 – glutPrint(float x, float y, void\* font, char\* text, float r, float g, float b, float a)

화면 내에 점수와 남은 시간, 스테이지 레벨과 아이템 보유상황을 표시하기 위해서 텍스트로 표시해줘야 한다. 이는 glutPrint함수가 담당한다. 좌우반전 시 텍스트가 깨지는 문제점을 방지하기 위해 rotation이 true,false 일 때 각각 x,y,z좌표를 설정하였다.

**Specification**

STL vector: 공,벽돌,아이템, 바운더리 같이 여러 개의 도형으로 구현해야하는 경우에 vector 클래스를 사용하였다. Initialize 함수 상단에 글로벌 변수로 선언하였다.

Class inheritance: Rectangle, Circle클래스가 Shape, Color 클래스를 상속하도록 구현하였다.

Operator overlading: Vec3f클래스를 생성하고 클래스함수로 연산자오버로딩함수를 구현하여 충돌함수에서 내적, 상수곱 같은 계산을 쉽게 하였다.

Dynamic allocation: 여러 도형들을 동적할당하였다. Vector에 집어넣을 때도 동적할당 후 집어넣었다. 물체를 없앨 때 delete를 하여 Memory leak를 방지하도록 노력하였다.

Function 요구사항은 위에 설명하였습니다.

**Feedback**

실제 게임으로 출시하기 위해서는 아직 구현해야 할 것이 많다. 우선 pause와 restart를 만들어 게임 재시작 및 정지를 불편함 없이 게임 내에서 처리하도록 해야한다. 또한 Highscore를 파일로 저장, 불러오기를 이용해 구현하여 게임에서 도전하기 위한 목표로 설정해야 한다. 만약 가능하다면 bmp 덮어쓰기를 통해 벽돌과 아이템, 공, 바를 좀더 세련되게 바꾸고, GUI를 배운 후 메뉴를 구현할 수 있으면 좋겠다고 생각한다. 아이템이 너무 많아질 것 같아서 추가로 넣지는 않았지만 바의 길이를 늘린다던가, 공을 아래로 떨어뜨려도 한번에 gameover되게 하지 않고 life를 도입해 일정 횟수만큼 재도전을 할 수 있게 한다던가, 블록을 1줄 파괴할 때마다 위에서 랜덤하게 배열된 벽돌이 내려와 무한히 벽돌을 부숴 score를 경쟁하는 무한 모드도 만들고 싶다.