**게임프로그래밍 기말프로젝트**

**과목명**  게임프로그래밍

**담당교수**  계희원

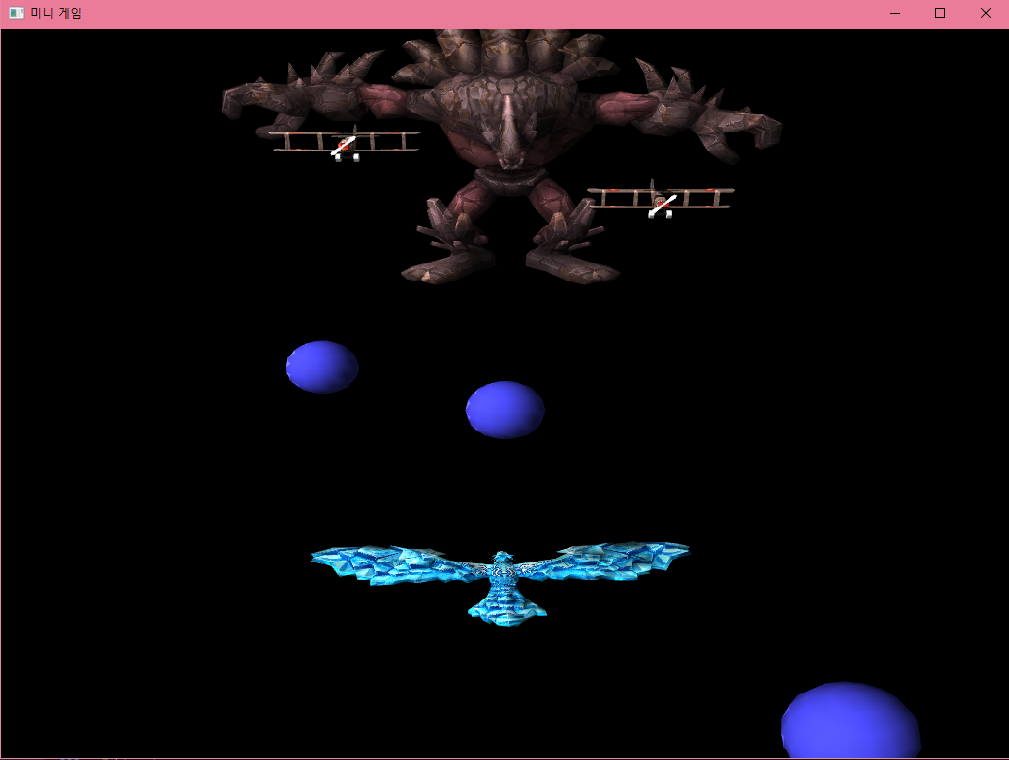
**학과명**  컴퓨터공학부

**학번**  1692074

**이름**  조현지



[1] 소개



새 캐릭터를 조종해 날라오는 공을 튕겨내 몬스터를 잡는 미니게임 입니다.

[2] 조작법

1)키보드

좌/우 키로 새 캐릭터를 조종합니다.

스페이스바로 현재 남은 HP와 적 HP를 확인 할 수 있습니다.

ESC키로 게임을 종료 할 수 있습니다.

2)마우스

왼쪽 버튼 클릭 후 마우스를 드래그 하면 화면을 움직일 수 있습니다.

오른쪽 버튼 클릭하면 멈춥니다.

[3] 구현

1) 모델링과 Mesh

새 모델과 적 모델, 공 모델 3개가 등장합니다.

/\*float 인자 3개는 R,G,B를 나타냅니다.

ball을 파랗게 만들기 위해 함수에 매개 변수를 추가하여 변경했습니다.\*/

g\_XFile.XFileLoad(g\_pd3dDevice, "./images/av.x",1.0f,1.0f,1.0f);

g\_XBall1.XFileLoad(g\_pd3dDevice, "./images/ball.x",0.3f, 0.3f, 1.0f);

g\_XBall2.XFileLoad(g\_pd3dDevice, "./images/ball.x", 0.3f, 0.3f, 1.0f);

g\_XBall3.XFileLoad(g\_pd3dDevice, "./images/ball.x", 0.3f, 0.3f, 1.0f);

g\_XEnemy.XFileLoad(g\_pd3dDevice, "./images/malp.x", 1.0f, 1.0f, 1.0f);

2) 배경

몬스터와 비행기 2개가 고정되어 있습니다.

3) 사용자 입력에 반응

// 키보드 입력

case WM\_KEYDOWN:

switch (wParam)

{

case VK\_SPACE:

TCHAR msg[100];

sprintf\_s(msg,"내 HP %d 적 HP %d", g\_PhyObject.HPCheck(),g\_PhyEnemy.HPCheck());

MessageBox(NULL, msg, "HP 확인", MB\_OK);

break;

case VK\_LEFT:

g\_PhyObject.SetXposition(-100);

break;

case VK\_RIGHT:

g\_PhyObject.SetXposition(100);

break;

case VK\_ESCAPE:

if (MessageBox(NULL, "게임을 종료 하시겠습니까?", "게임 종료 창", MB\_YESNO) == IDYES)

exit(0);

break;

}

break;

// 마우스 입력 (카메라)

case WM\_LBUTTONDOWN:

move = true;

break;

case WM\_MOUSEMOVE:

if(move)

if (LOWORD(lParam) >= 504 && HIWORD(lParam) >= 377) {

g\_pCamera->MoveLocalX(2.5f);

g\_pCamera->MoveLocalY(-2.5f);

}

else if (LOWORD(lParam) >= 504 && HIWORD(lParam) <= 377) {

g\_pCamera->MoveLocalX(2.5f);

g\_pCamera->MoveLocalY(2.5f);

}

else if (LOWORD(lParam) <= 504 && HIWORD(lParam) >= 377) {

g\_pCamera->MoveLocalX(-2.5f);

g\_pCamera->MoveLocalY(-2.5f);

}

else {

g\_pCamera->MoveLocalX(-2.5f);

g\_pCamera->MoveLocalY(2.5f);

}

break;

case WM\_RBUTTONDOWN:

move = false;

break;

}

4) 충돌 검사

// Collision 함수 : 새와 공 충돌 함수

// 왼쪽 공과 오른쪽 공은 공 스피드에 따라 v.x값을 다르게 주었습니다.

// z값은 세 공 모두 새와 반대 방향으로 가게끔 -v.z값을 주었습니다.

// 왼쪽 공

if(rsum > length){

if (p.x <= -100) {

if (target->v.z <= -14.0f) {

target->v.x = 2.7;

}

else if (target->v.z <= -13.0f) {

target->v.x = 2.6;

}

else if (target->v.z <= -12.0f) {

target->v.x = 2.4;

}

else if (target->v.z <= -11.0f) {

target->v.x = 2.2;

}

else if (target->v.z <= -10.0f) {

target->v.x = 2.0;

}

else if (target->v.z <= -9.0f) {

target->v.x = 1.8;

}

else if (target->v.z <= -8.0f) {

target->v.x = 1.6;

}

else if (target->v.z <= -7.0f) {

target->v.x = 1.4;

}

else if (target->v.z <= -6.0f) {

target->v.x = 1.2;

}

else {

target->v.x = 1.0;

}

target->v.z = -target->v.z;

}

// 오른쪽 공

else if(p.x >= 100) {

if (target->v.z <= -15.0f) {

target->v.x = -2.7;

}

else if (target->v.z <= -14.0f) {

target->v.x = -2.5;

}

else if (target->v.z <= -13.0f) {

target->v.x = -2.4;

}

else if (target->v.z <= -12.0f) {

target->v.x = -2.2;

}

else if (target->v.z <= -11.0f) {

target->v.x = -2.1;

}

else if (target->v.z <= -10.0f) {

target->v.x = -1.9;

}

else if (target->v.z <= -9.0f) {

target->v.x = -1.7;

}

else if (target->v.z <= -8.0f) {

target->v.x = -1.5;

}

else if (target->v.z <= -7.0f) {

target->v.x = -1.3;

}

else if (target->v.z <= -6.0f){ // 충돌 3

target->v.x = -1.1;

}else{ // 충돌 3

target->v.x = -0.9;

}

target->v.z = -target->v.z;

}

else {

target->v.z = -target->v.z;

}

}

//CollisionEnemy 함수 : 몬스터와 공 충돌 함수

if (rsum >= length) {

n\_collision++; // 충돌 횟수 증가

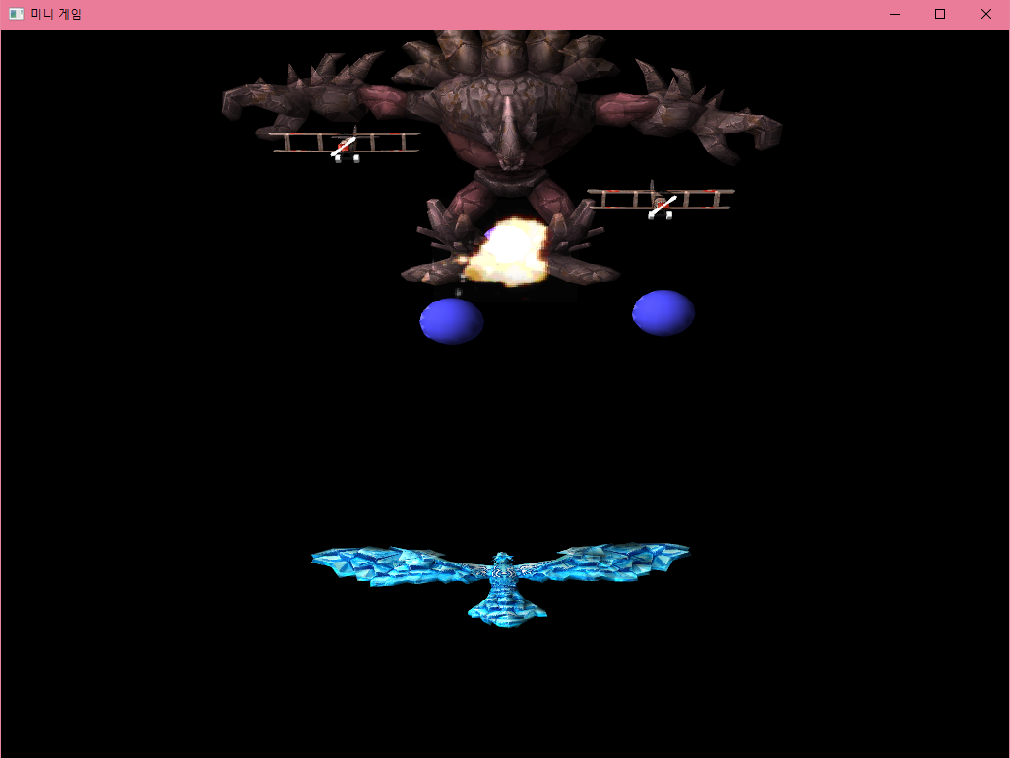
}

// 충돌횟수가 1이상일 때 화염 효과 그림

if (g\_PhyEnemy.NumberCollision()>=1) {

g\_pd3dDevice->DrawPrimitive(D3DPT\_TRIANGLESTRIP, 0, 2);

}



5) 화면 구성

공은 스피드가 랜덤입니다.

int PhysicalObj::RandomSpeed() {

srand(time(NULL));

return rand() % 11 + 5; // 5~15

}

VOID Render()

if (g\_PhyEnemy.HPCheck() <= 0) { // 적 HP가 0이되면

TCHAR msg[100];

sprintf\_s(msg, "적 처치 성공! 남은 HP %d", g\_PhyObject.HPCheck());

if (MessageBox(NULL, msg, "GAME SUCCESS", MB\_OK) == IDOK) {

exit(0); // 플레이어 남은 HP를 알려주는 메세지박스

}

}

if (g\_PhyObject.HPCheck() <= 0) { // 플레이어 HP가 0이되면

TCHAR msg[100];

sprintf\_s(msg, "남은 적 HP %d", g\_PhyEnemy.HPCheck());

if (MessageBox(NULL, msg, "GAME OVER", MB\_OK) == IDOK) {

exit(0); // 적의 남은 HP를 알려주는 메세지박스

}

}

// ball1

matWorld = g\_PhyBall1.GetWorldMatrix();

g\_pd3dDevice->SetTransform(D3DTS\_WORLD, &matWorld);

if (g\_PhyBall1.Zposition() < -400) { // 못 맞추고 새 뒤로 넘어 갔을 때 초기화

g\_PhyObject.HPReduction(1); // 플레이어 HP 감소

g\_PhyBall1.SetPosition(-100, 0, 180); // 위치 초기화

g\_PhyBall1.SetVelocity(0, 0, -g\_PhyBall1.RandomSpeed()); // 스피드 랜덤 초기화

}

if (g\_PhyBall1.Zposition() > 500) { // 맞췄을 때 초기화

g\_PhyEnemy.HPReduction(1); // 적 HP 감소

g\_PhyEnemy.SetCollision(g\_PhyEnemy.NumberCollision()); // 충돌 횟수 초기화

g\_PhyBall1.SetPosition(-100, 0, 180);// 위치 초기화

g\_PhyBall1.SetVelocity(0, 0, -g\_PhyBall1.RandomSpeed()); // 스피드 랜덤 초기화

}

g\_XBall1.XFileDisplay(g\_pd3dDevice);