3D게임프로그래밍2 (01)

과제2 설명

제출일: 2023.12.19

학과: 게임공학과

학번: 2019184020

성명: 윤은지

목차

1. 조작법(키)에 대한 설명

1)플레이어

```
462  void CGameFramework::ProcessInput()
463  {
464      static UCHAR pKeysBuffer[256];
465      bool bProcessedByScene = false;
466      if (GetKeyboardState(pKeysBuffer) && m_pScene) bProcessedByScene = m_pScene->ProcessInput(pKeysBuffer);
467      if (!bProcessedByScene)
468      {
469          DWORD dwDirection{};
470          if (pKeysBuffer[0x57] & 0xF0) dwDirection |= DIR_FORWARD;//W
471          if (pKeysBuffer[0x53] & 0xF0) dwDirection |= DIR_BACKWARD;//S
472          if (pKeysBuffer[0x41] & 0xF0) dwDirection |= DIR_LEFT;//A
473          if (pKeysBuffer[0x44] & 0xF0) dwDirection |= DIR_RIGHT;//D
474          if (pKeysBuffer[0x58] & 0xF0) dwDirection |= DIR_UP;//X
475          if (pKeysBuffer[0x43] & 0xF0) dwDirection |= DIR_DOWN;//C
476          if (pKeysBuffer[0x10] & 0xF0 && dwDirection) dwDirection |= DIR_RUN;// Shift run
```

w : 앞으로 이동

s : 뒤로 이동

a : 왼쪽으로 이동

d : 오른쪽으로 이동

x : 위로 이동

c : 아래로 이동

SHIFT : 가속

```
287  void CGameFramework::OnProcessingMouseMessage(HWND hWnd, UINT nMessageID, WPARAM wParam, LPARAM lParam)
288  {
289      random_device rd;
290      mt19937 gen(rd());
291      uniform_int_distribution<> dist(3, 5);
292
293      if (m_pScene) m_pScene->OnProcessingMouseMessage(hWnd, nMessageID, wParam, lParam);
294      switch (nMessageID)
295      {
296      case WM_LBUTTONDOWN:
297          ::SetCapture(hWnd);
298          ::GetCursorPos(&m_ptOldCursorPos);
299          break;
300      case WM_RBUTTONDOWN:
301
302          m_pScene->pMultiSpriteObjectShader->m_ppObjects[0]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
303          m_pPlayer->attack = true;
304
305
306          m_pScene->pMultiSpriteObjectShader->score = dist(gen);
307          cout << m_pScene->pMultiSpriteObjectShader->score << endl;
308
309          break;
```

마우스 우클릭 : 공격

2)적 헬리콥터

```
392  bool CScene::OnProcessingKeyboardMessage(HWND hWnd, UINT nMessageID, WPARAM wParam, LPARAM lParam)
393  {
394      switch (nMessageID)
395      {
396      case WM_KEYDOWN:
397          switch (wParam)
398          {
399          case VK_UP:
400
401              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
402                  pObjectsShader->m_ppObjects[i]->MoveForward(+1.0f);
403              break;
404          case VK_DOWN:
405
406              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
407                  pObjectsShader->m_ppObjects[i]->MoveForward(-1.0f);
408              break;
409          case VK_LEFT:
410
411              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
412                  pObjectsShader->m_ppObjects[i]->MoveStrafe(-1.0f);
413              break;
414          case VK_RIGHT:
415
416              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
417                  pObjectsShader->m_ppObjects[i]->MoveStrafe(+1.0f);
418              break;
419          case VK_RETURN:
420
421              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
422                  pObjectsShader->m_ppObjects[i]->MoveUp(+1.0f);
423              break;
424          case 0x10://SHIFT
425
426              for (int i{}; i < pObjectsShader->m_nObjects; ++i)
427                  pObjectsShader->m_ppObjects[i]->MoveUp(-1.0f);
428              break;
```

UP : 앞으로 이동

DOWN : 뒤로 이동

LEFT : 왼쪽으로 이동

RIGHT : 오른쪽으로 이동

ENTER : 위로 이동

SHIFT : 아래로 이동

3)테셀레이션

```
case '1':
    m_pcbMappedFrameworkInfo->m_nRenderMode = 0x00;
    break;
case '2':
    m_pcbMappedFrameworkInfo->m_nRenderMode |= DYNAMIC_TESSELLATION;
    break;
case '3':
    m_pcbMappedFrameworkInfo->m_nRenderMode  |= (DYNAMIC_TESSELLATION | DEBUG_TESSELLATION);
    break;
case '4':
    ::gbTerrainTessellationWireframe = !::gbTerrainTessellationWireframe;
    break;
```

숫자 1, 2, 3, 4 : 테셀레이션 렌더 모드 전환

4)시작 화면

마우스 좌클릭 : 위의 메뉴를 클릭

```
firstAssignment                                   CGameFramework
327  void CGameFramework::OnProcessingKeyboardMessage(HWND hWnd, UINT nMessageID, WPARAM wParam, LPARAM lParam)
328  {
329      if (m_pScene) m_pScene->OnProcessingKeyboardMessage(hWnd, nMessageID, wParam, lParam);
330      switch (nMessageID)
331      {
332      case WM_KEYUP:
333          switch (wParam)
334          {
335          case VK_ESCAPE:
336              if (onFullScreen)
337                  ChangeSwapChainState();
338
339              ::PostQuitMessage(0);
340              break;
341          case VK_RETURN:
342              break;
343          case VK_F1:
344          case VK_F2:
345          case VK_F3:
346              m_pCamera = m_pPlayer->ChangeCamera((DWORD)(wParam - VK_F1 + 1), m_GameTimer.GetTimeElapsed());
347              break;
348          case VK_CONTROL:
349              if (onFullScreen)
350                  onFullScreen = false;
351              else
352                  onFullScreen = true;
353              ChangeSwapChainState();
```

ESC : 종료

CTRL : 전체 화면으로 전환

또한 Release/x64 모드를 사용하여 프로젝트를 빌드하였습니다.

2. 게임 규칙

1) 게임 클리어 조건

```
if (crashCnt == pObjectsShader->m_nObjects) {
    sound[0].Stop();//??????
    sound[2].Stop();
    sound[4].Play();//?????

    pMultiSpriteObjectShader->m_ppObjects[7]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
    pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
    start = false;
}
```

약 60~90초의 제한 시간 안에 적 헬리콥터 24대를 총으로 맞히면 게임 클리어

2) 게임 오버 조건

```
if (0 < m_pPlayer->HP) {
    //m_pPlayer->HP -= 1.5314f;
    //m_pPlayer->HP -= 0.09259f;
    m_pPlayer->HP -= 0.00009259f;
    m_pPlayer->Render(pd3dCommandList, pCamera);
}
else {
    pMultiSpriteObjectShader->m_ppObjects[8]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
    pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
    sound[0].Stop();//??????
    sound[3].Play();//?????
}
```

제한 시간 안에 적 헬리콥터를 모두 맞히지 못하면 게임 오버

3. 과제1 프로젝트에 거울, 테셀레이션 프로젝트 통합

Shaders.hlsl에 다음을 추가하였습니다.

```
struct VS_LIGHTING_INPUT

struct VS_LIGHTING_OUTPUT

VS_LIGHTING_OUTPUT     VSCubeMapping(VS_LIGHTING_INPUT input)

float4   PSCubeMapping(VS_LIGHTING_OUTPUT   input)   :
```

SV_Target

TextureCube gtxtCubeMap : register(t1);

Shader.h에 다음을 추가하였습니다.

class CDynamicCubeMappingShader : public CTexturedShader

class CTerrainTessellationShader : public CShader

Object.h에 다음을 추가하였습니다.

class CDynamicCubeMappingObject : public CGameObject

Mesh.h에 다음을 추가하였습니다.

class CMeshIlluminated : public CMesh

class CDiffused2TexturedVertex : public CDiffusedVertex

1) 거울

```
void CScene::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList)
```

위의 함수에서

```
m_nEnvironmentMappingShaders = 1;
m_ppEnvironmentMappingShaders = new CDynamicCubeMappingShader * [m_nEnvironmentMappingShaders];

m_ppEnvironmentMappingShaders[0] = new CDynamicCubeMappingShader(256);
m_ppEnvironmentMappingShaders[0]->CreateShader(pd3dDevice, pd3dCommandList, m_pd3dGraphicsRootSignature);
m_ppEnvironmentMappingShaders[0]->BuildObjects(pd3dDevice, pd3dCommandList, m_pTerrain);
```

거울 구를 생성합니다.

```
void CScene::OnPreRender(ID3D12Device* pd3dDevice, ID3D12CommandQueue* pd3dCommandQueue, ID3D12Fence* pd3dFence, HANDLE hFenceEvent)
{
    for (int i = 0; i < m_nEnvironmentMappingShaders; i++)
    {
        m_ppEnvironmentMappingShaders[i]->OnPreRender(pd3dDevice, pd3dCommandQueue, pd3dFence, hFenceEvent, this);
    }
}
```

위의 함수에서 거울에 비칠 씬을 렌더합니다.

```
void CGameFramework::FrameAdvance()
```

위의 함수에서

```
m_pScene->OnPreRender(m_pd3dDevice, m_pd3dCommandQueue, m_pd3dFence, m_hFenceEvent);
```

거울에 비칠 씬을 렌더하는 함수를 호출합니다.

```
void CScene::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera)
```

위의 함수에서 아래와 같이 거울 구를 렌더합니다.

```
for (int i{}; i < m_nEnvironmentMappingShaders; ++i)
    m_ppEnvironmentMappingShaders[i]->Render(pd3dCommandList, pCamera);
```

2) 테셀레이션

```
void CScene::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList)
```

위의 함수에서

```
m_pTerrain = new CHeightMapTerrain(pd3dDevice, pd3dCommandList, m_pd3dGraphicsRootSignature, _T("Image/HeightMap.raw"), 257, 257, 13,
```

동적lod를 테셀레이션으로 구현한 지형을 생성합니다.

```
void CScene::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera)
```

위의 함수에서

```
if (m_pTerrain) m_pTerrain->Render(pd3dCommandList, pCamera);
```

지형을 렌더합니다.

```
case '1':
    m_pcbMappedFrameworkInfo->m_nRenderMode = 0x00;
    break;
case '2':
    m_pcbMappedFrameworkInfo->m_nRenderMode |= DYNAMIC_TESSELLATION;
    break;
case '3':
    m_pcbMappedFrameworkInfo->m_nRenderMode  |= (DYNAMIC_TESSELLATION | DEBUG_TESSELLATION);
    break;
case '4':
    ::gbTerrainTessellationWireframe = !::gbTerrainTessellationWireframe;
    break;
```

숫자 키 1, 2, 3, 4에 따라 변하는 테셀레이션 렌더 모드 변수입니다.

```
void CGameFramework::FrameAdvance()
```

위의 함수에서

```
m_pScene->x = &m_pcbMappedFrameworkInfo->m_nRenderMode;
```

테셀레이션 렌더 모드 변수를 넘기고

```
void CScene::UpdateShaderVariables(ID3D12GraphicsCommandList* pd3dCommandList)
{
    ::memcpy(m_pcbMappedLights->m_pLights, m_pLights, sizeof(LIGHT) * m_nLights);
    ::memcpy(&m_pcbMappedLights->m_xmf4GlobalAmbient, &m_xmf4GlobalAmbient, sizeof(XMFLOAT4));
    ::memcpy(&m_pcbMappedLights->m_nLights, &m_nLights, sizeof(int));
    ::memcpy(&m_pcbMappedLights->renderMode, x, sizeof(UINT));
}
```

위의 함수에서 셰이더로 테셀리이션 렌더 모드 변수를 넘겨줍니다.

```
float4 PSTerrainTessellation(DS_TERRAIN_TESSELLATION_OUTPUT input) : SV_TARGET
{
    float4 cColor = float4(0.0f, 0.0f, 0.0f, 1.0f);

    if (gnRenderMode& (DEBUG_TESSELLATION | DYNAMIC_TESSELLATION))
    {
```

위와 같이 테셀레이션 렌더 모드 변수에 따라 if조건문의 코드 블록이 실행될지 안 될지 결정되고 지형이 다르게 그려집니다.

4. UI

1) 시작, 종료 화면

```
void CMultiSpriteObjectsShader::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList, void* pContext)
```

위의 함수에

```
ppSpriteTextures[6] = new CTexture(1, RESOURCE_TEXTURE2D, 0, 1);
ppSpriteTextures[6]->LoadTextureFromDDSFile(pd3dDevice, pd3dCommandList, L"Image/start.dds", RESOURCE_TEXTURE2D, 0);

ppSpriteTextures[7] = new CTexture(1, RESOURCE_TEXTURE2D, 0, 1);
ppSpriteTextures[7]->LoadTextureFromDDSFile(pd3dDevice, pd3dCommandList, L"Image/win.dds", RESOURCE_TEXTURE2D, 0);

ppSpriteTextures[8] = new CTexture(1, RESOURCE_TEXTURE2D, 0, 1);
ppSpriteTextures[8]->LoadTextureFromDDSFile(pd3dDevice, pd3dCommandList, L"Image/over.dds", RESOURCE_TEXTURE2D, 0);
```

위와 같이 시작, 종료 화면의 텍스처를 로드 하였습니다.

```
else if(9!=j)//화면
    pSpriteMesh = new CTexturedRectMesh(pd3dDevice, pd3dCommandList, 130.0f, 10.0f, 0.0f, 0.0f, 45.0f, 0.0f);
```

메쉬는 위와 같이 생성하였습니다.

```
else if(9!=i){
    xmf3PlayerPosition.x = (xmf3PlayerPosition.x + 1.f * xmf3CameraPosition.x) / 2.f;
    xmf3PlayerPosition.y = (xmf3PlayerPosition.y + 1.f * xmf3CameraPosition.y) / 2.f;
    xmf3PlayerPosition.z = (xmf3PlayerPosition.z + 1.f * xmf3CameraPosition.z) / 2.f;
}

if (9 != i) {
    m_ppObjects[i]->SetPosition(xmf3PlayerPosition);
    m_ppObjects[i]->SetLookAt(xmf3CameraPosition, XMFLOAT3(0.0f, 1.0f, 0.0f));
}
```

플레이어와 카메라와의 거리는 위와 같이 설정하여 빌보드가 화면을 향하게 하였습니다.

```
void CScene::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList)
```

위의 함수에서

```
pMultiSpriteObjectShader->m_ppObjects[6]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
```

시작 화면을 렌더하기 위로 true로 설정하고

```
void CGameFramework::FrameAdvance()
```

위의 함수에서

```
if (false == onFullScreen){
    if (446 <= m_ptOldCursorPos.x - windowX && 513 >= m_ptOldCursorPos.x - windowX
        && 359 <= m_ptOldCursorPos.y - windowY && 393 >= m_ptOldCursorPos.y - windowY){//play
        ShowCursor(false);
        m_pScene->start = true;
        m_pScene->pMultiSpriteObjectShader->m_ppObjects[6]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
        m_pScene->sound[1].Stop();//??????
        m_pScene->sound[0].Play();//?????

    }
```

Play 문구가 있는 위치를 마우스 좌 클릭하면 시작 화면을 false로 설정하였습니다.

```
void CScene::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera)
```

위의 함수에서

```
if (start) {

    if (0 < m_pPlayer->HP) {
        //m_pPlayer->HP -= 1.5314f;
        m_pPlayer->HP -= 0.09259f;
        //m_pPlayer->HP -= 0.00009259f;
        m_pPlayer->Render(pd3dCommandList, pCamera);
    }
    else {
        pMultiSpriteObjectShader->m_ppObjects[8]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
        pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
        start = false;
        sound[0].Stop();//??????
        sound[3].Play();//?????
    }

}
```

플레이어의 체력 변수가 0보다 작으면 즉, 제한 시간이 지나면 game over 화면을 true로 설정하였습니다.

```
void CScene::AnimateObjects(float fTimeElapsed)
```

위의 함수에서

```
if (crashCnt == pObjectsShader->m_nObjects) {
    sound[0].Stop();//??????
    sound[2].Stop();
    sound[5].Stop();
    sound[4].Play();//?????

    pMultiSpriteObjectShader->m_ppObjects[7]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
    pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
    start = false;
}
```

적 헬리콥터와 총알의 충돌 횟수가 적 헬리콥터의 수와 같으면 game clear 화면을 true로 설정하였습니다.

2) 총 조준선

```
void CMultiSpriteObjectsShader::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList, void* pContext)
```

위의 함수에

```
ppSpriteTextures[9] = new CTexture(1, RESOURCE_TEXTURE2D, 0, 1);
ppSpriteTextures[9]->LoadTextureFromDDSFile(pd3dDevice, pd3dCommandList, L"Image/crosshair.dds", RESOURCE_TEXTURE2D, 0);
```

위와 같이 총 조준선 텍스처를 로드 하였습니다.

```
else//조준
    pSpriteMesh = new CTexturedRectMesh(pd3dDevice, pd3dCommandList, 3.0f, 3.0f, 0.0f, 0.0f, 0.0f, 0.0f);
```

메쉬는 위와 같이 생성하였습니다.

```
else if(9==j)
    pSpriteObject->m_ppMaterials[0]->m_pTexture->texMat.z = 2;
```

위와 같이 셰이더에 넘기는 변수의 값을 2로 할당하여

```
float4 PSTextured(VS_TEXTURED_OUTPUT input) : SV_TARGET
{
    float4 cColor = gtxtTexture.Sample(gssWrap, input.uv);

    if (gMaterial.texMat.z == 2 && cColor.x < 0.4f)
        discard;

    return(cColor);
}
```

총 조준선 텍스처의 배경을 투명하게 하였습니다.

```
else {

    xmf3PlayerLook = pPlayer->GetLookVector();

    xmf3Position = Vector3::Add(xmf3PlayerPosition, Vector3::ScalarProduct(xmf3PlayerLook, 150.0f, false));

    m_ppObjects[i]->SetPosition(xmf3Position);
    m_ppObjects[i]->SetLookAt(xmf3CameraPosition, XMFLOAT3(0.0f, 1.0f, 0.0f));
}
```

플레이어와 카메라의 위치 정보를 이용하여 계산한 값으로 총 조준선 빌보드가 플레이어 앞의 특정 위치에 그려지도록 했습니다.

```
void CScene::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera)
```

위의 함수에서

```
if (pCamera->GetPlayer() && start) {
    m_pShadowMapToViewport[1]->Render(pd3dCommandList, pCamera, 5400 / 25.f, XMFLOAT2(30, 21));
    m_pShadowMapToViewport[0]->Render(pd3dCommandList, pCamera, m_pPlayer->HP / 25.f, XMFLOAT2(38, 27));
    pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
}
```

pCamera->GetPlayer()는 총 조준선이 거울에 비치지 않게 하기 위함이고, start 변수는 인 게임 진입 여부를 나타냅니다. If 조건문이 참이면 총 조준선을 렌더할지 결정하는 변수에 true를 할당하였습니다.

3) 체력 바

Shader.h에

```
class CTextureToViewportShader : public CShader
```

위의 클래스를 추가하고

Shader.cpp에 클래스의 구현부를 추가하였습니다.

```
void CScene::BuildObjects(ID3D12Device* pd3dDevice, ID3D12GraphicsCommandList* pd3dCommandList)
```

위의 함수에서

```
m_pShadowMapToViewport = new CTextureToViewportShader * [2];
m_pShadowMapToViewport[0] = new CTextureToViewportShader();
m_pShadowMapToViewport[0]->CreateShader(pd3dDevice, pd3dCommandList, m_pd3dGraphicsRootSignature);

m_pShadowMapToViewport[1] = new CTextureToViewportShader();
m_pShadowMapToViewport[1]->CreateShader(pd3dDevice, pd3dCommandList, m_pd3dGraphicsRootSignature);
m_pShadowMapToViewport[1]->color = 1;
```

체력 바의 배열 사이즈를 2로 하고, 0번 인덱스에는 제한 시간을 의미하는 빨간 색의 체력을 할당하고, 1번 인덱스에는 체력 바의 배경을 나타내는 민트 색의 뒷판을 할당하였습니다.

color 변수는 0으로 초기화 되어있고,

```
float4 PSTextureToViewport(VS_TEXTURED_OUTPUT input) : SV_Target
{
    if(gMaterial.hpColor==0)
        return float4(1.f,0.f,0.f,0.f);
    else
        return float4(0.f,1.f,1.f,0.f);
}
```

위의 픽셀 셰이더에서 color 변수 값에 따라 빨강 혹은 민트 색을 반환합니다.

```
void CScene::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera)
```

위의 함수에서

```
if (pCamera->GetPlayer() && start) {
    m_pShadowMapToViewport[1]->Render(pd3dCommandList, pCamera, 5400 / 25.f, XMFLOAT2(30, 21));
    m_pShadowMapToViewport[0]->Render(pd3dCommandList, pCamera, m_pPlayer->HP / 25.f, XMFLOAT2(38, 27));
    pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
}
```

위와 같이 체력 바를 렌더합니다.

체력 바의 뒷판의 크기는 5400으로 고정이고, 빨간 색 체력은 m_pPlayer->HP 변수 값에 따라 변하는데 m_pPlayer->HP의 초기값은 5000입니다.

```
if (start) {

    if (0 < m_pPlayer->HP) {
        //m_pPlayer->HP -= 1.5314f;
        m_pPlayer->HP -= 0.09259f;
        //m_pPlayer->HP -= 0.00009259f;
        m_pPlayer->Render(pd3dCommandList, pCamera);
    }
    else {
        pMultiSpriteObjectShader->m_ppObjects[8]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
        pMultiSpriteObjectShader->m_ppObjects[9]->m_ppMaterials[0]->m_pTexture->m_bActive = false;
        start = false;
        sound[0].Stop();//??????
        sound[3].Play();//?????
    }

}
```

위와 같이 CScene::Render 함수 호출 시 0.09259f 만큼을 m_pPlayer->HP 변수에서 뺍니다. 인 게임에 진입하고 약 60~90초가 경과하면 m_pPlayer->HP 변수가 0보다 작아지게 됩니다.

```
void CTextureToViewportShader::Render(ID3D12GraphicsCommandList* pd3dCommandList, CCamera* pCamera, float plHp, XMFLOAT2 pos)
{

    D3D12_RECT d3dScissorRect = { pos.x,pos.y, pos.x + plHp, pos.y + 15.f };

    if (5400/25.f == plHp)
        d3dScissorRect.bottom += 12;

    pd3dCommandList->RSSetScissorRects(1, &d3dScissorRect);

    CShader::Render(pd3dCommandList, pCamera);

    pd3dCommandList->SetGraphicsRoot32BitConstants(1, 1, &color, 29);

    pd3dCommandList->IASetPrimitiveTopology(D3D_PRIMITIVE_TOPOLOGY_TRIANGLELIST);
    pd3dCommandList->DrawInstanced(6, 1, 0, 0);
}
```

위와 같이 체력 바가 그려질 위치를 지정한 후 color 변수를 셰이더에 넘기고 체력 바를 그렸습니다.

5. 사운드

stdafx.h에 다음 라이브러리를 추가하였습니다.

```
//sound
#include <xaudio2.h>
#pragma comment(lib, "xaudio2.lib")
```

Scene.h에 다음 구조체와 클래스를 추가하고, Scene.cpp에 다음 클래스의 구현부를 추가하였습니다.

```
41    struct WAVEHEADER

58    class SoundPlayer
```

CScene 클래스에 다음 변수들을 추가하였습니다.

```
SoundPlayer sound[6];
const wchar_t* inGame = _T("Sound/inGame.wav");
const wchar_t* opening = _T("Sound/opening.wav");
const wchar_t* att = _T("Sound/gun.wav");
const wchar_t* win = _T("Sound/win.wav");

const wchar_t* monster = _T("Sound/hit.wav");
const wchar_t* close = _T("Sound/closing.wav");
```

1) 배경음

```
void CScene::BuildObjects(ID3D12Device* pd
{

    // Initialize SoundPlayer
    sound[0].Initialize();
    sound[0].LoadWave(inGame, 0);


    sound[1].Initialize();
    sound[1].LoadWave(opening, 0);

    sound[3].Initialize();
    sound[3].LoadWave(close, 0);

    sound[4].Initialize();
    sound[4].LoadWave(win, 0);
```

위와 같이 오프닝, 게임 오버, 게임 클리어, 인 게임 배경음을 초기화 하였습니다.

2) 플레이어의 총 발사음

```
case WM_RBUTTONDOWN:

    m_pScene->pMultiSpriteObjectShader->m_ppObjects[0]->m_ppMaterials[0]->m_pTexture->m_bActive = true;
    m_pPlayer->attack = true;

    m_pScene->pMultiSpriteObjectShader->score = dist(gen);
    cout << m_pScene->pMultiSpriteObjectShader->score << endl;

    if (!m_pScene->sound[2].attOnce) {
        m_pScene->sound[2].Initialize();
        m_pScene->sound[2].LoadWave(m_pScene->att, 0);
        m_pScene->sound[2].Play();//?????

        m_pScene->sound[2].attOnce = true;
    }
```

위와 같이 마우스 우 클릭을 하면 총 발사음이 재생되도록 하였습니다.

3) 적 헬리콥터 피격음

```
if (!pObjectsShader->obj.empty()) {
    for (int i{}; i < pObjectsShader->obj.size(); ++i)
    {
        float bullet_monster_distance = Vector3::Length(Vector3::Subtract(pObjectsShader->obj[i]->aabb.Center, Cur_Pos));

        if (pObjectsShader->obj[i]->aabb.Intersects(Bullet_Origin, Bullet_Direction, bullet_monster_distance))
        {

            cout << i << "명중" << endl;
            pMultiSpriteObjectShader->hit = pObjectsShader->obj[i]->GetPosition();
            pObjectsShader->obj.erase(pObjectsShader->obj.begin() + i);
            cout << "obj.size : " << pObjectsShader->obj.size() << endl;
            cout << "obj.i : " << i << endl;
            m_pPlayer->attack = false;
            ++crashCnt;
            if (!sound[5].attOnce) {
                sound[5].Initialize();
                sound[5].LoadWave(monster, 0);
                sound[5].Play();

                sound[5].attOnce = true;
            }
```

위와 같이 총알의 바운딩 박스와 적 헬리콥터의 바운딩 박스가 충돌하면 피격음이 재생되도록 하였습니다.

6. 실행 화면

1) 시작 화면

2) 인 게임

3) 게임 오버

4) 게임 클리어


GAME CLEAR