* **[조작법]**

|  |  |
| --- | --- |
| **키** | **설명** |
| **W, A, S, D** | **이동** |
| **Space / LShift** | **탄 발사 / 부스터** |
| **F2 / F3** | **자유 시점(플레이어를 그리지 않는다) / 플레이어 시점** |
| **R / F** | **상승 / 하강.** |
| **ESC** | **프로그램 종료** |

* **[사용한 자료구조]**

**Scene**

**RootParameter**

[0]: CBV – register b0 // player

-**RootDescRange**

Range[0]: CBV (b2) // objects

numDesc - 1

Range[1]: SRV (t0) // terrain

numDesc - 3

Range[2]: SRV (t3) // skybox

numDesc - 6

Range[3]: SRV (t9) // billboard

numDesc - 6

Range[4]: SRV (t15) // particle

numDesc - 1

[1]: CBV – register b1 // camera

[2]: Table – range 1 // objects

[3]: CBV – register b2 // materials

[4]: CBV – register b3 // lights

[5]: Table – range 1 // terrain

[6]: Table – range 1 // skybox

[7]: Table – range 1 // billboard

[8]: Table – range 1 // particle

-**RootDescRange**

Range[0]: CBV (b5) //

numDesc - 1

Range[1]: SRV (t16) // texture

numDesc - 2

**PostProcessingShader**

**RootParameter**

[1]: Table – range 1 // RenderTarget

[0]: Table – range 1 // PostProcessingInfo

* 씬의 루트 시그니처는 과제 3과 동일하고 미니맵 및 블러효과를 주기 위한 셰이더클래스에 루트 시그니처를 하나 더 추가하였다.
* **[과제 3에서 구현하지 못한 내용 보충]**

1. **적 오브젝트와 플레이어 오브젝트의 위치를 나타내는 미니맵을 구현한다.**

* 가정

1. 8-9-1 샘플 프로젝트의 포스트 프로세싱(외곽선 그리기)을 위한 멀티 렌더타겟을 사용하는 샘플코드를 기반으로 한다.
2. 스왑체인 외에 렌더타겟을 2개 더 사용하여 한곳에는 씬을 플레이어 시점으로 그리고 다른곳에는 미니맵을 직각투영으로 그려서 출력할 때 두 텍스처를 합쳐서 출력한다.
3. 미니맵은 별도의 카메라를 만들어서 구현한다.

* 구현

1. PostProcessingShader클래스를 만들고 루트 시그니처를 [사용한 자료구조]와 같은 형태로 만든다.
2. .프레임워크에서 8-9-1 샘플 프로젝트의 렌더타겟을 여러 개 생성하는 코드를 참조하여 FrameAdvanced() 함수를 수정하고, PostProcessingShader 클래스를 TextureToFullScreenByLaplacianShader클래스의 내용으로 채우고 CB\_POSTPROCESSING\_INFO구조체를 만들고 CBV버퍼를 생성하는 부분과 Render할 때 루트 시그니처에 Set하는 부분을 추가한다.
3. 카메라 클래스에 프로젝션 행렬을 orthographic으로 만들어주는 함수를 추가하고 프레임워크의 멤버변수로 minimap 카메라를 추가한 후 buildobject() 함수에서 minimap 카메라를 viewport를 조정하여 오른쪽 하단에 200x200 크기로 출력하도록 만들고, projection행렬을 orthographic으로 만들어서 직각 투영이 되게 하고, 지형 중앙에서 y가 2000인 위치에 minimap 카메라를 set position해준후에 카메라가 지면을 바라보도록 만들었다.
4. Shaders.hlsl코드의 Laplacian 픽셀셰이더 함수의 내용을 다 지우고 이름을 byLaplacian에서 withPostProcessing으로 바꾼 후 gtxtScene(t1), gtxtNormal(t2) texture2D들을 gtxtScene(t16), gtxtMiniMap(t17)로 바꾼 후 프레임워크의 FrameAdvanced()에서 렌더타겟들의 clear color에서 알파 값을 0으로 설정하고 OMSet(rendertarget[0]) -> 플레이어 카메라로 씬, 플레이어 그리기 -> OMSet(rendertarget[1]) -> 적 오브젝트와 플레이어 10배 확대 -> minimap 카메라로 씬 그리기 -> 적 오브젝트와 플레이어 0.1배 축소 -> PostProcessingShader Render() -> present() 순으로 그리기를 하도록 수정한다.
5. PSTextureToFullScreenWithPostProcessing 셰이더 함수에서 gtxtScene의 알파 값이 0에 근접하면 MiniMap을 출력하게 한다.
6. **적 오브젝트는 미사일을 쏘며 플레이어가 맞으면 폭발 파티클 애니메이션이 실행된다.**

* 가정

1. 과제 3에서 플레이어가 헬기를 공격했을 때 폭발하는 파티클 애니메이션을 구현했던 방식으로 캐릭터가 공격을 받았을 때의 파티클 애니메이션을 구현한다.
2. 헬기는 미사일을 한발 씩 쏘며 플레이어와 미사일의 거리가 가까우면 플레이어는 미사일과 충돌한다.
3. 플레이어가 미사일에 맞으면 1초 후 지형 중앙에서 되살아난다.

* 구현

1. 건쉽 오브젝트에 발사를 나타내는 bool 변수를 하나 추가한후, Scene에서 플레이어와의 거리가 150 미만일때 look 벡터와 50\*timeElapsed를 곱한 만큼의 거리를 미사일을 헬파이어미사일프레임 변수를 이용하여 미사일의 위치를 이동시켰다.
2. 미사일 발사 bool변수가 true일 경우 건쉽 오브젝트의 world행렬을 업데이트 하지 않도록 해서 미사일이 날아가는 동안은 헬기가 움직이지 않도록 했다.
3. 미사일이 헬기에서 150이상 떨어지거나 플레이어에 충돌하면 미사일 발사 bool 변수가 false가 되고 미사일이 폭발하고 빌보드파티클애니메이션이 실행된다.
4. 헬기가 플레이어와 100 미만의 거리에 있으면 미사일을 발사한다.
5. 헬기가 플레이어와 100 이상 떨어져 있으면, 헬기는 3초마다 방향을 바꾸며 천천히 주변을 배회한다.

* **[프로그램 요구사항]**

1. **기하 셰이더로 빌보드를 구현한다.**

* 구현

1. 과제 2, 3번 모두 기하 셰이더를 이용해 빌보드를 구현하였다.
2. **플레이어에 부스터 기능을 추가하고 부스터를 사용중에는 블러 효과를 준다.**

* 가정

1. 미니맵을 구현할 때 사용했던 PostProcessingShader클래스와 픽셀 셰이더 코드에서 블러효과를 구현한다.
2. PostProcessingShader클래스의 CB\_POSTPROCESSING\_INFO구조체를 이용하여 부스터 키 입력을 셰이더 코드에서 알 수 있도록 한다.
3. PSTextureToFullScreenWithPostProcessing셰이더 함수에서 부스터 키가 입력되었을 경우에 블러효과를 적용한다.
4. 블러효과는 화면 중앙부근에는 적용되지 않는다..

* 구현

1. CB\_POSTPROCESSING\_INFO의 멤버변수로 UINT m\_bActive를 추가하고 매 render때마다 비디오메모리 값이 갱신되도록 하였다.
2. Shift키가 눌리면 구조체의 m\_bActive 변수가 1이되고 PSTextureToFullScreenWithPostProcessing 함수에서 블러기능이 활성화된다.
3. 블러링 패치는 3x3의 크기이고 각각의 원소는 1/9 이다.
4. 외곽선그리기 코드와 같은 방식으로 블러패치를 회선 처리하였고 position.xy를 if문으로 하드코딩하여 화면 중앙에는 블러연산을 수행하지 않도록 하였다.
5. Shift키가 눌리면 플레이어의 MaxVelocity가 800이 되고 Move의 인자가 50에서 200으로 커지게 하였다.
6. Shift키가 눌리지 않고 있을 때 MaxVelocity가 400, Move의 인자가 50으로 설정된다.



미사일 쏘는 헬기

우 하단 미니맵

부스터 및 블러 효과