|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **주차** | 12 주차 | **기간** | 2020.03.16 ~ 2020.03.22 | **지도교수 이용희** | (서명) |
| 이번주 한일 요약 | - 미사일 발사 구현(유도X)  - 경계구 충돌 테스트  - Compute Shader를 이용한 Blur 효과, Blur on / off 기능 구현 | | | | |

<상세 수행내용>

**이재원 :**

1. 전투기 모델에 붙어있는 미사일을 공격키 입력 시 실시간으로 동적으로 생성되게 하기 위하여 우선 3DMAX로 미사일 모델만 따로 FBX파일로 익스포트 후 유니티를 통하여 bin파일로 재 추출(프레임워크에서 불러오기 위함)

실외, 비행, 평면, 비행기이(가) 표시된 사진

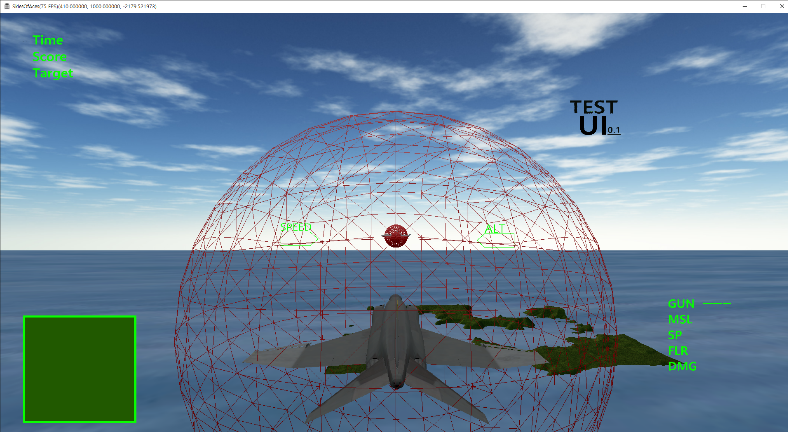
자동 생성된 설명

2. 플레이어 클래스에서 추출된 미사일 모델을 미리 임포트 후 스페이스(임시)를 누를 때 마다 미사일 클래스를 동적 생성하여 모델을 적용시킴

그 후 미사일 클래스 내에서 생성될 때 받아온 매개변수인 유도될 오브젝트의 좌표를 받아 저장하고 경계구를 생성, 매프레임마다 Look방향으로 직진하는 함수를 작성함

발사되는 것을 확인하기 위하여 발사되는 미사일의 크기로 임의로 수정함

또한 충돌 테스트를 테스트씬 클래스 내에서 하였지만 효율적인 충돌체크를 위하여 CollisionManager클래스를 작성할 계획임



(경계구 메쉬를 렌더한 상태로 미사일을 발사한 장면)



(경계구 메쉬를 렌더하지 않고 발사한 장면1)



(경계구 메쉬를 렌더하지 않고 발사한 장면2)

미사일에 씌워진 경계구와 적 오브젝의 경계구가 충돌하는 것을 확인하였음.

그러나 생성된 첫번째 미사일만 충돌함으로 충돌 매니저 클래스 작성이 필요함

**김령운 :**

가속도에 의한 의식상실을 시각적으로 구현하기 위해 블러 효과를 구현했다.

1. 장면을 화면 밖 텍스처에 그린다

- 장면을 후면 버퍼에 렌더링할 때에는 먼저 다음과 같이 후면 버퍼의 렌더 타겟 뷰를 렌더링 파이프라인의 출력병합기 단계에 묶는다.

m\_pd3dCommandList->OMSetRenderTargets(1, &d3dRtvCPUDescriptorHandle, TRUE, &d3dDsvCPUDescriptorHandle);

이후 IDXGISSwapChain::Present 메서드를 호출해서 후면 버퍼를 "제시(presenting)"하면 결과적으로 후면 버퍼 내용이 화면에 나온다.

장면을 후면 버퍼가 아닌 화면 밖 텍스처에 그리기 위해서 화면 밖 텍스처를 후면 버퍼와 같은 형식, 같은 크기로 생성하여 후면 버퍼를 렌더링한 후 CopyResource를 이용해 화면 밖 텍스처에 복사했다.

2. 화면 밖 텍스처를 Compute Shader 프로그램을 이용하여 흐린다.

- 흐리기 가중치 평균 계산을 위해서 가우스 흐리기(Gaussian blur) 연산자에서 사용하는 가우스 함수로 구한 가중치들을 사용했다.

텍스처 표본 추출은 비싼 연산이므로, 흐리기를 분리해서 텍스처 표본 개수를 줄이면 성능에 이점이 있다. 성능을 위해 분리 가능이라 가정하고 흐리기 연산을 단순화했다(수학적으로 가우스 흐리기는 분리 가능 흐리기에 속한다).

2차원 흐리기가 분리 가능이라고 간주하고 두 개의 1차원 흐리기 (수직, 수평)로 분리해서 계산을 수행했다.

Blur 구현을 위한 텍스처 버퍼 두 개가 필요하며 두 텍스처 모두 읽기와 쓰기가 가능해야 한다. 따라서 두 텍스처 각각에 대해 SRV와 UAV를 만들었다. 한 텍스처를 A, 다른 한 텍스처를 B라고 부르기로 하겠다. 흐리기 알고리즘은 다음과 같이 진행된다.

1. A에 대한 SRV를 Compute Shader의 입력으로 설정(수평으로 흐릴 입력 이미지)

2. B에 대한 UAV를 Compute Shader의 출력으로 설정(흐려진 결과를 담을 출력 이미지)

3. 스레드 그룹들을 배분해서 수평 흐리기 작업을 수행, 작업이 모두 끝나면 텍스처B는 수평으로 흐려진 결과인 BlurH(I)를 담은 상태(I는 흐리고자 하는 원본 이미지)

4. B에 대한 SRV를 Compute Shader의 입력으로 설정(수직으로 흐릴 입력 이미지)

5. A에 대한 UAV를 Compute Shader의 출력으로 설정(흐려진 결과를 담을 출력 이미지)

6. 스레드 그룹들을 배분해서 수직 흐리기 작업을 수행, 작업이 모두 끝나면 텍스처A는 수직으로 흐려진 결과인 BlurV(I)를 담은 상태(I는 흐리고자 하는 원본 이미지)

텍스처 A 와 B가 교대로 Compute Shader의 입출력으로 쓰이지만, 동시에 입력이자 출력으로 쓰이지 않는다. 수평, 수직 흐리기 패스가 결합되면 하나의 온전한 흐리기 패스가 된다. 흐려진 결과를 또 다른 흐리기 패스를 적용해 더 흐리는 것도 가능하다.

[Compute Shader에서 픽셀 흐리기]

- 입력 이미지의 너비가 w 이고 높이가 h라고 가정하고 이미지의 모든 픽셀을 흐리려면 수평 1차원 흐리기의 경우 x방향으로 ceil(올림) w / N 만큼, y 방향으로 h 개의 스레드 그룹을 배분해야 한다.

각 방향으로 배분할 스레드 그룹 개수를 파악하고, 흐리기 연산을 위한 Compute Shader 호출 명령을 등록하는 메서드는 루나 예제에 있는 BlurFilter::Execute 메서드를 참고하여 만들었다.

만약 N이 w로 나누어 떨어 지지 않는 다면 스레드 그룹의 마지막에 여분의 스레드들이 생긴다. 스레드 그룹의 크기는 고정되어 있으므로 이는 어쩔 수 없는 일이다.

범위 밖 색인으로 픽셀을 읽는 것이 오류는 아니고 0을 돌려주도록 정의되어 있지만 0은 RGB가 모두 0인 검은색을 의미하기 때문에 흐리기에 관여하게 되면 이미지의 가장자리가 어두워진다.

이런 현상을 해결하기 위해 텍스처 좌표 지정 모드 중 하나인 clamp 모드를 적용할 때처럼 범위 밖 색인에 대해서는 범위 안 텍셀과 동일한 값을 돌려주게 했다. 이렇게 하면 제일 오른쪽 스레드 그룹에는 출력 텍스처의 그 어떤 텍스처 와도 대응되지 않는 여분의 스레드들이 되고, 범위 밖 쓰기가 일어나 아무 일도 일어나지 않는다.

[공유 메모리 사용]

텍스처의 표본화는 GPU연산들 중 상대적으로 느린 연산에 속하기 때문에 스레드 그룹에 필요한 텍스처 표본들을 공유 메모리의 배열에 미리 담아두고 중복된 텍스처 추출을 피할 수 있다. 매번 텍스처를 추출하는 대신 공유 메모리 배열에 담긴 표본을 조회하여 성능이 크게 향상되도록 하였다.

1. 문제점

한 스레드 그룹의 스레드가 n개이고 흐리기 반지름이 R이라고 할 때 하나의 스레드가 흐리기를 수행하려면 n+ 2R개의 텍셀을 읽어야 한다. 흐리기 반지름 때문에 이미지 가장자리 픽셀들에 대한 스레드들은 스레드 그룹의 바깥의 유효하지 않은 픽셀을 참조하게 된다.

2. 해결책

공유 메모리에 n + 2R의 항목을 할당하고 2R개의 스레드가 텍셀 값 두 개를 조회하게 하였다.

[주의점]

스레드 그룹의 모든 스레드가 동시에 작업을 진행한다는 보장이 없기에 한 스레드가 미처 공유 메모리 배열의 한 원소를 초기화 하기 전에 다른 한 스레드가 그 원소에 접근하는 일이 얼마든지 일어날 수 있다.

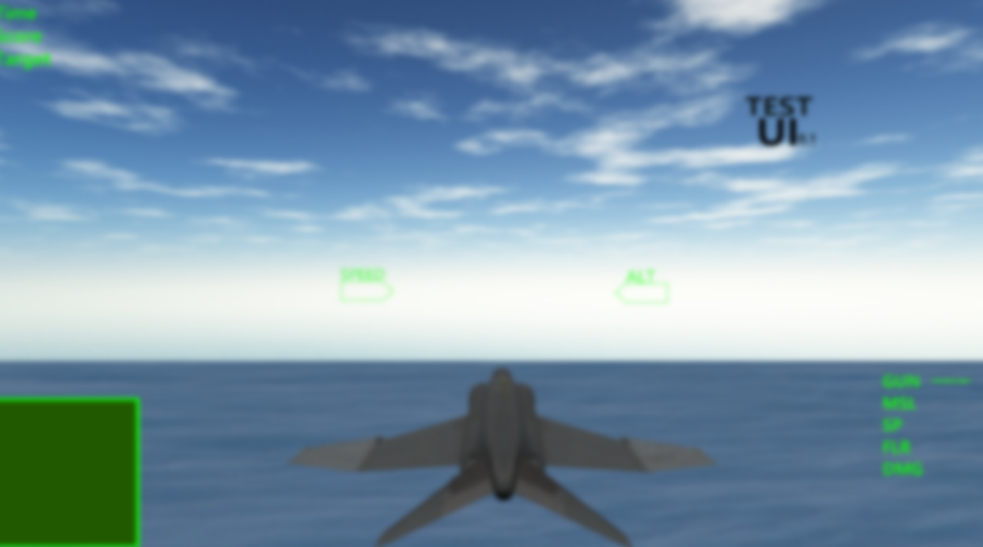
이를 방지하기 위해 동기화 함수를 사용했다. (GroupMemoryBarrierWithGroupSync())

3. 후면 버퍼를 다시 렌더 대상으로 설정하고, 흐려진 텍스처를 화면 전체 크기의 사각형에 입힌다.

m\_pd3dCommandList->CopyResource(CurrentBackBuffer(), m\_pBlurFilter->Output());



흐리기 전



흐리기 후

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** | 미사일이 발사는 되나 유도기능을 구현하는데 오류가 있음 | | |
| **해결방안** | 기존에 과제에서 썼던 SetLookAt함수를 참고, 수정하여 부드러운 유도기능을 추가 | | |
| **다음주차** | 13 주차 | **다음기간** | 2020.03.23 ~ 2020.03.29 |
| **다음주 할일** | 이재원 : 미사일 유도기능 구현, 빌보드 애니메이션 구현, 충돌매니저 클래스 구현  김령운 : 스프라이트 애니메이션, 미니맵 제작 | | |
| **지도 교수**  **Comment** |  | | |