**목표 : 3D게임1 01 폴더의 내용 (윈도우 프로그래밍으로 작성된 코드)를 Directx 12 API를 사용하게 바꾸는 것**

**조작법**

상하좌우 방향키로 플레이어이를 이동

스페이스키로 총알 발사

마우스로카메라의 회전 (왼쪽 클릭하면서 움직여야 합니다.)

F1, F2, F3로 시점 변경

**충돌**

충돌 부분에서 플레이어와 다른 객체들과 충돌(벽, 장애물)하기 위해 OOBB를 설정해 놓았는데 **3D게임1 01 폴더**에서 한 내용을 그대로 OOBB를 transform하게 되면 벽 충돌 할 때 문제가 있었습니다.

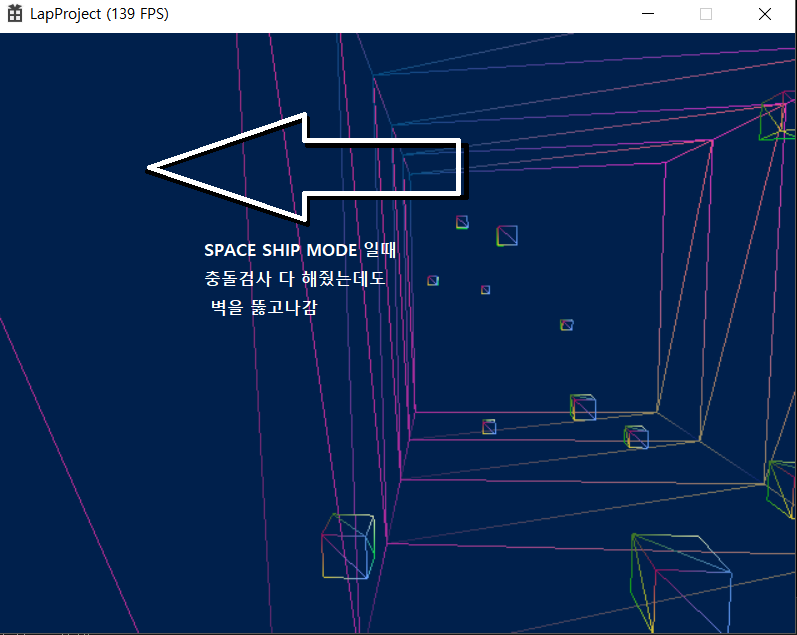
예를 들어

m\_pPlayer->m\_xmOOBBTransformed.Transform(m\_pPlayer->m\_xmOOBB, XMLoadFloat4x4(&m\_pPlayer->m\_xmf4x4World));

이런식으로 이동시키면 m\_xmf4X4Xworld 행렬이 3인칭일때는 잘 변환 하다가 스페이스 쉽이모드나 1인칭 모드로 바꾸면 단위행렬로 초기화 되었습니다..

처음에는 위의 사실을 확인 못해서 보는 시점(1인칭,3인칭)이 달라 졌을때마다 Case를 나눠서 충돌검사를 해야하나 생각하였습니다.

Transform 함수에 인자로 들어가는 m\_pPlayer->m\_xmf4X4World가 단위행렬로 변하니 충돌검사가 잘 이루어지지 않았습니다.



그래서 그러면 애초에 카메라에 oobb를 달아주면 카메라 시점이 바뀌어도 잘 동작할거라 믿고 바꿔서 계속 구현해 나가다가**(GameFramework.cpp -551줄)** m\_pPlayer가 Camera를 가지고 있어서 굳이 이렇게 해야하나 생각이 들었고,

Transfrom에 인자로 넣어주는 월드변환을 플레이어의 Right, Up, Look, Position벡터 하나하나 써주면 시점이 바뀐다고해서 월드변환 행렬이 변하지 않을꺼라 생각하여서 하나하나 인자로 채워넣어서 구현하였습니다..

**관련된 코드는 GameFramework.cpp 545줄에 있음**

**총알 생성**

처음 총알 클래스를 게임프레임워크에 ProcessInput에 따라 생성을 해주려 하였지만,

총알생성

게임 프레임워크

씬

메쉬

(RENDER)

게임오브젝트

쉐이더

총알을 GameFrameWork에서 생성하면 어떻게 그리는지에 대해서 많은 시간이 걸렸습니다. 생성된 총알을 프레임워크에서 단순히 Render함수만 호출하면 바로 그릴 수 있다고 잘못 생각하였습니다.

문제를 해결하기 위해 쉐이더에 총알을 선언하고 Buildobject에서 생성하였습니다. 그리고 GameFramework에서 space 키 입력 있을때마다 Shader.h에서 static변수로 bShootState변수를 선언해 놓고 true이면 Shader에 있는 총알이 Animate 되게 구현하였습니다.

이 부분이 **3D게임1 02**를 진행하며 상당히 아쉬운 부분이고 static변수 선언하는 것도 C++ 객체지향에 맞게 Set함수로 자식클래스까지 인자를 전달하고 싶었지만 그렇게 하지 못하였던 것도 아쉬웠습니다 (이 문제는 3D게임1 03에서 해결하였습니다.). Player의 값도 Shader에서 처리하기위해 static변수로 선언하였습니다. (이 부분은 과제3 진행하면서 C++ 관련 서적 참고해서 플레이어는 GameFramework ->Scene에 Set함수 (자식 클래스까지 다) 선언해서 구현하였음) – **관련내용은 GameFramework.cpp -466줄, 359줄에**

**실행결과**

총알의 OOBB를 설정하고 Animate 해주고 하는 부분들은 과제1과 동일합니다.

총알에 부딪히는 객체는 Z좌표를 증가시켜 뒤로 보내게 하였고, 장애물과 플레이어가 충돌하면 플레이어의 좌표를 원점으로 Set해주었습니다.

3D게임1 01과 다르게 총알의 자료구조는 배열을 사용하였습니다. (총알은 총 32발)

그 외에 플레이어와 장애물간의 충돌, 장애물끼리의 충돌 부분은 3D게임1 01과 동일하게 구현하였습니다.