설명 : 미로에서 장애물을 피해 탈출하는 게임

**조작법**

방향키로 이동하고

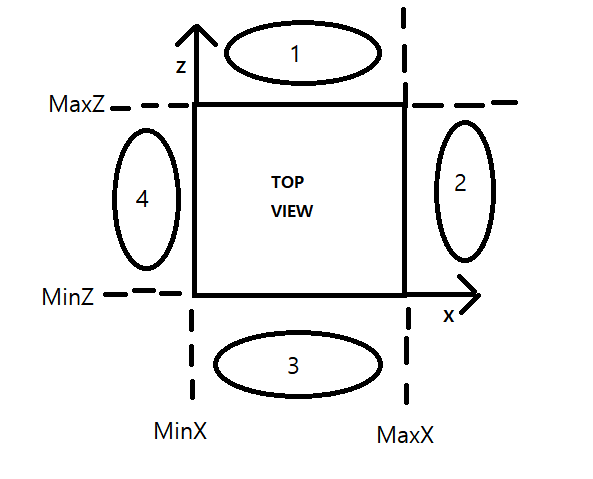
P키 – 미로 앞으로 이동

장애물(구)은 마우스 클릭으로 없앨 수 있습니다.(피킹 구현)

마우스로 회전 (마우스 왼쪽클릭 해야합니다.)

**목표**

벽들 충돌체크



벽은 직육면체이고 하늘에서 바라보면 위와 같은 그림이 됩니다.

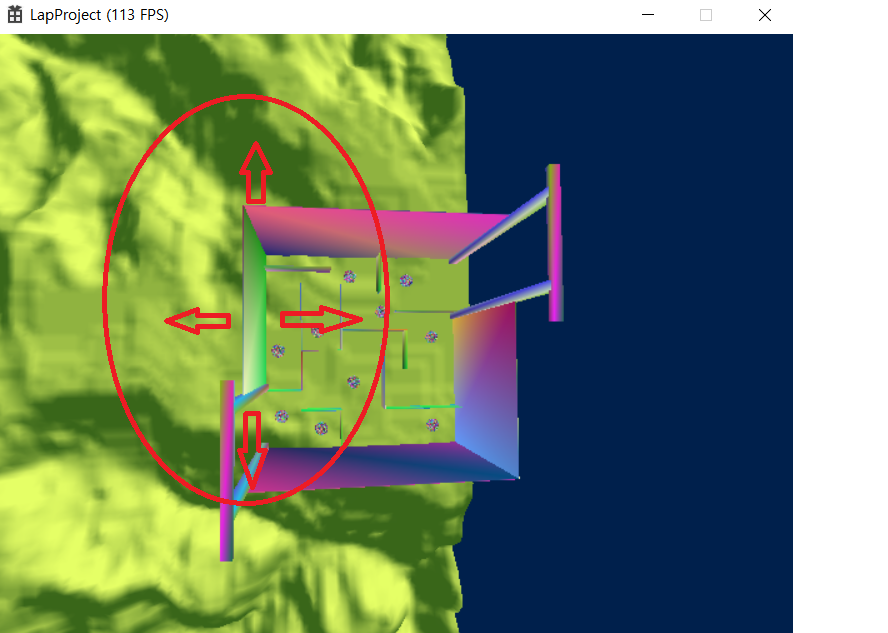
1번 영역 : (p(플레이어)의 Z좌표 > MaxZ) && (p의 X좌표 > MinX) && (p의 X좌표 < MaxX)

2번 영역 : (p의 x좌표 > MaxX) && (p의 z좌표 > MinZ) && (p의 z좌표 < MaxZ)

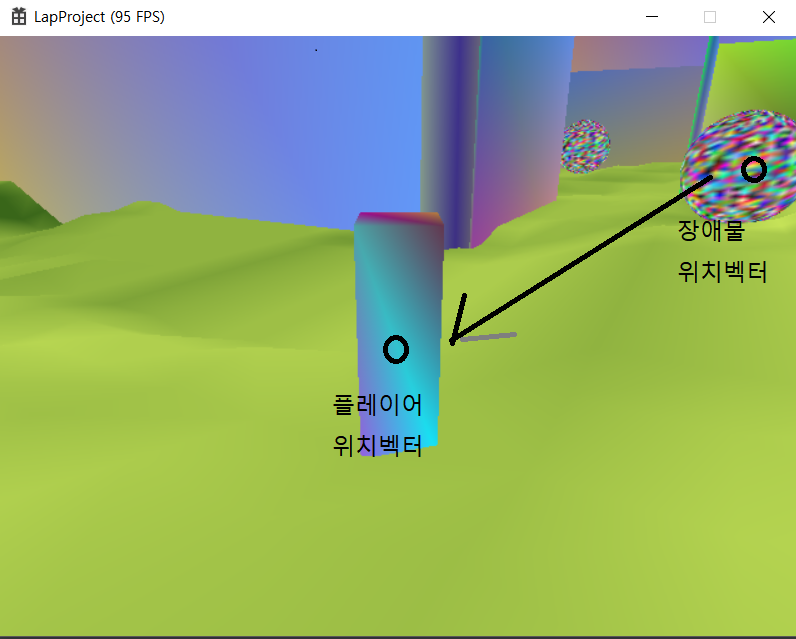
3번 영역 : (p의 z좌표 < MinZ) && (p의 x좌표 > MinX) && (p의 x좌표 < MaxX)

4번 영역 : (p의 x좌표 < MinX) && (p의 z좌표 > MinZ) && (p의 z좌표 < MaxZ)

그래서 만약 벽의 oobb와 플레이어의 oobb가 intersects 한 값이 존재 한다면 위와 같은 식으로 충돌처리를 할 수 있습니다. 충돌을 하면 각 1,2,3,4 면 중 하나에 해당하는 노멀벡터로 플레이어를 밀어내도록 하였습니다. **(관련코드 shader.cpp – 950줄)**



구와 플레이어가 일정 거리가 되면 장애물은 플레이어를 향해 오도록 구현 하였습니다.

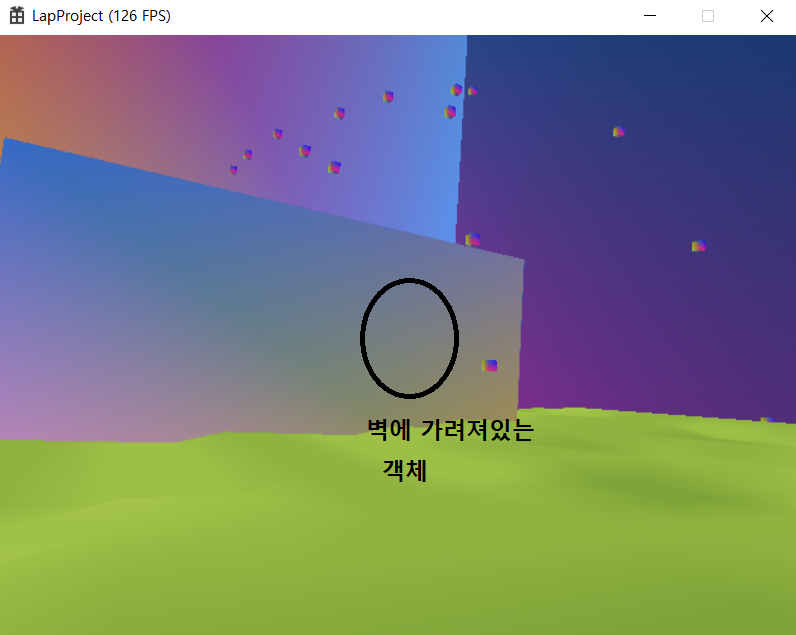


플레이어의 oobb의 Center와 (위치벡터와 같음) 장애물의 oobb의 Center를 점과 점사의 거리 공식을 사용하여 일정거리 이하가 되었을 때 구를 움직이도록 구현하였습니다.

구를 움직이기 위해선 위의 그림과 같은 화살표 방향으로의 방향벡터가 필요한데 이는 구의 위치벡터와 플레이어의 위치벡터의 차를 이용하여 구하였습니다. 이렇게 구한 방향벡터를 속도를 곱한값을 플레이어의 위치에 더해주었습니다. **(관련코드 – shader.cpp 1063줄)**

**피킹**

적을 없애는데 피킹을 사용하면서 고려해야할 점은 적이 벽에 가려져 있는 경우입니다.



벽에 구가 가려져 있는 상황을 고려하지 않으면 벽에 가려 보이지 않는 객체도 없앨 수 있게 되는 것입니다. 따라서 마우스를 클릭할 때 평면이 있는지 확인을 하고, 만약 평면이 존재한다면 그 평면과 카메라의 거리가 피킹할 구와 카메라의 거리보다 크면, (크다는 것은 더 멀리있단 소리니까) 그때서야 객체를 없애도록 구현하였습니다. **(관련코드 - shader.cpp 799줄)**