2013182017 김희동

키 설정

WASD(전후좌우) QE(상하)

Z : 30m 반경 적 제거 아이템 사용

(플레이어 좌표와 오브젝트 좌표로 백터 생성하여 길이 비교)

SPACE : 피킹플래그 (활설화/비활성화) + 마우스클릭 = 피킹

활성화된 상태에서 마우스로 피킹 가능

활성화/비활성화시 누르면 비활성화/활성화

R : 게임 재시작

G : 피킹해제

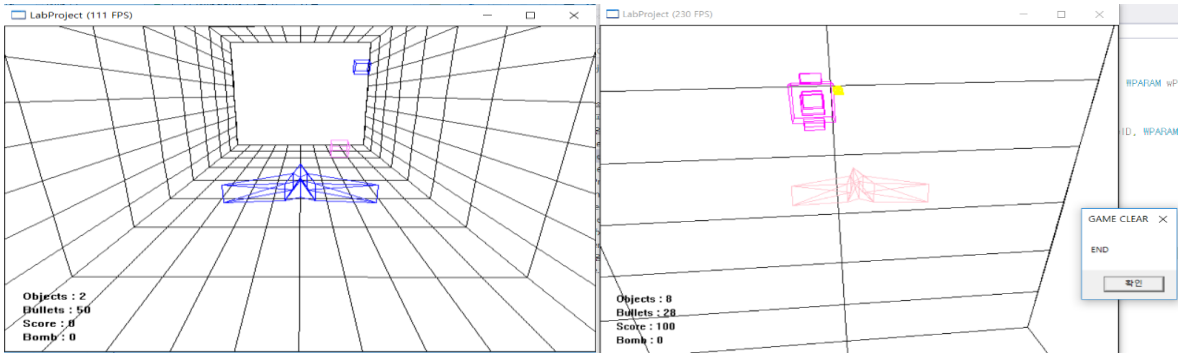
T : 모든 적 제거(폭발 일어나지 않음)

Ctrl키 : 총알 발사

마우스 왼쪽버튼 드래그 : x,y 축회전

마우스 오른쪽버튼 드래그 : x,z 축회전

시작/끝 화면



가로 300, 세로 300 길이 1000의 맵, 한쪽 끝에 보스

좌측하단에 textout함수를 이용하여 남은 총알 수, 점수, 아이템 개수, 살아있는 오브젝트 수 표기

보스가 사망시 messagebox함수를 이용하여 end 출력 후 종료

충돌처리

플레이어와 오브젝트, 총알, 보스, 벽의 충돌처리

=플레이어의 위치(점)과 OBB의 충돌검사함수(Contains)를 사용하여 OBB에 포함되었거나 교차했을 경우 충돌처리(게임 재시작)

=플레이어와 벽의 충돌처리의 경우, 움직일 때마다 충돌검사를 하여 Contains일 경우 움직인 방향을 기억하여, Contains가 아닐 경우 움직인 방향의 반대로 충분한 거리만큼 move해준다.

오브젝트와 총알

=OBB끼리의 충돌검사함수(Intersects)를 사용하여 교차일 경우 충돌처리(점수, 특정오브젝트의 경우 아이템 획득, 렌더플래그 활성화)

총알과 벽

=OBB끼리 충돌검사함수(Contains)를 사용하여 벽OBB에 포함되지 않을 경우 충돌처리 (렌더플래그 활성화)

보스, 오브젝트와 벽

=벽 OBB에 포함되는 지 체크(Contains), 교차했거나 포함되지 않았을 경우 Intersects함수를 이용하여 포함되지 않았다면 어떤 면뒤에 있는지 파악, 교차했을 경우 교차한 면을 파악하여 반사벡터를 구해서 이동방향에 적용

오브젝트와 오브젝트

OBB끼리 충돌검사함수(Intersects)를 이용하여 교차했을 경우 서로의 이동방향벡터와 속력을 교환해준다.

피킹

SPACE + 마우스 선택으로 피킹

G 피킹해제

Getcursor함수로 화면상 좌표를 받은 후, ScreenToCilent함수를 사용하여 클라이언트 좌표로 변환한다.

변환한 좌표를 월드좌표계까지 역변환을 해주어 나온 점과 카메라 위치(점)을 벡터를 생성하고 노멀화 하여 광선방향벡터를 생성한다.

카메라 위치를 시점으로 반복문을 돌리는 데 만약 해당 점이 보스 또는 오브젝트의 OBB에 포함되어 있지않으면 시점에 구한 방향벡터를 더해준다. 이렇게 반복하다보면 이 점은 광선벡터처럼 직선으로 나아간다. 그러다가 OBB에 포함되거나 카메라 위치부터 더해진 점까지 벡터의 길이가 200m가 넘어가면 탈출한다.

여기서 보스 피킹의 경우에는 피킹인덱스 값은 -2로, 오브젝트 피킹의 경우에는 오브젝트의 인덱스를, 초기값은 -1을 가지고 있는 피킹인덱스변수로 구분한다.

오브젝트의 경우 제일 앞에 있는 오브젝트에 피킹이 되며 보스와 오브젝트가 겹쳐있을 경우 보스 피킹이 우선적으로 된다.

피킹이 끝났을 경우 다시 좌표계를 화면좌표계로 바꿔준다.

피킹한 물체가 폭발했을 시 피킹은 해제된다.

피킹 후 바라보는 작업.

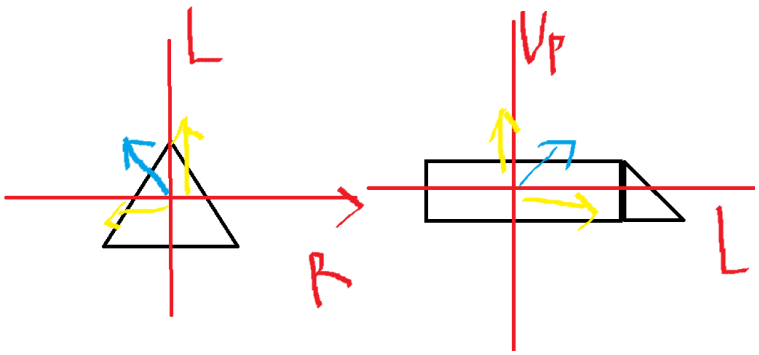
첫째, y축회전각을 알기 위하여 플레이어 y좌표와 오브젝트의 y좌표를 0으로 놓고 두 점을 빼기연산과 노멀화를 하여 플레이어로부터 오브젝트를 가리키는 방향벡터를 생성한다.

그렇게 생성된 방향벡터와 플레이어의 Look벡터를 내적하면, 연산의 사용되는 벡터가 둘다 단위벡터기 때문에 cos 값을 알 수 있다.

이 cos값에 acos을 취해줘서 y축회전각도를 구한다..

x축 역시 이와 같은 방법으로 x축회전각도를 구할 수 있다.

두번째로 각도를 구했으니 이것이 왼쪽으로 도는지, 오른쪽으로 도는지 파악을 하기위해 벡터의 덧셈을 이용하였는데, 플레이어 기준 왼쪽 벡터를 얻기 위해 Look벡터와 Right벡터의 반대방향 벡터를 덧셈으로 구하였다. 이러한 방법으로 좌우상하 대각선을 벡터를 구한다. (L : Look, R : Right)



그 다음 플레이어가 물체를 바라보는 방향의 벡터의 빼기 연산을 하여 거리를 비교 후 거리가 더 작은 쪽이 플레이어와 물체가 바라봐야 할 방향이라는 것을 알 수 있다.

보라색 : 플레이어가 오브젝트를 바라보는 벡터

파랑색 : 좌우 대각선 벡터

빨강색 : 플레이어가 오브젝트를 바라보는 벡터와 대각선 벡터를 뺄셈을 통해 얻은 벡터

빨강색의 크기가 작은 쪽으로 기울인다.

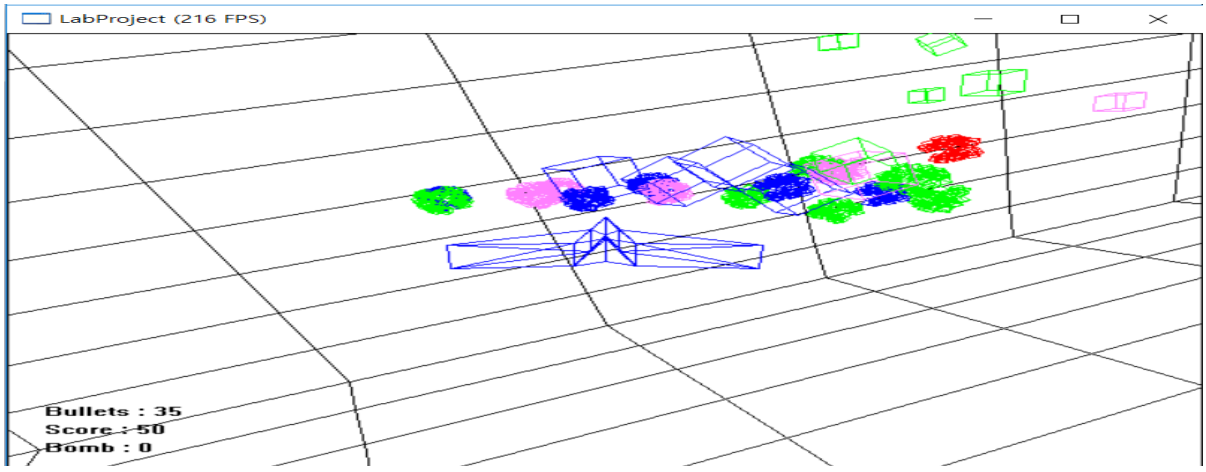
이렇게 얼마나 돌리는 지와 어디로 돌려야 하는지를 알아서 바라볼 수 있다.

오브젝트

프레임워크로 들어오면 시작시간을 재고 animate함수를 들어올 때의 시간을 체크하여 1초가 되면 오브젝트 생성함수를 호출하여 오브젝트의 값들을 세팅한다. 오브젝트 값들은 랜덤함수를 통해 세팅되며, 회전축에 경우 0,0,0이 될 수 없으므로 한 축에 1값을 고정으로 바꿔가며 넣어주었다.

플레이어의 z 좌표가 350미만일 경우 오브젝트는 100m,앞에서 그 외의 경우에는 z좌표가 400m로 지정해놓았다. 나머지 좌표는 랜덤 그대로이다.

이 함수에서는 빨강 오브젝트를 제외한 오브젝트만 생성하며, 이 함수가 10번(10초가 되었을 경우) 호출될 경우 빨강 오브젝트를 생성하는 함수를 호출한다. 또 플레이어와 오브젝트 위치간의 벡터를 통해 플레이어를 따라가게 만들어준다. 오브젝트는 생성시 1초동안 랜덤이동을 하다가 1초 후 플레이어를 쫓게된다.



오브젝트는 총알과 충돌할 시 100조각으로 터지며, 폭발 시 변수의 값에 따라 출력을 조정하여 파편이 깜빡이게 하였다. 폭발은 1초동안 이루어지고 폭발이 끝나면 오브젝트의 데스플래그와 폭발플래그를 사용하여 폭발시 그림은 그려지되, 충돌체크에서 빠지게 하였고, 폭발이 끝났으면 animate와, render 등 실시간처리에서 빠지게 하였다.

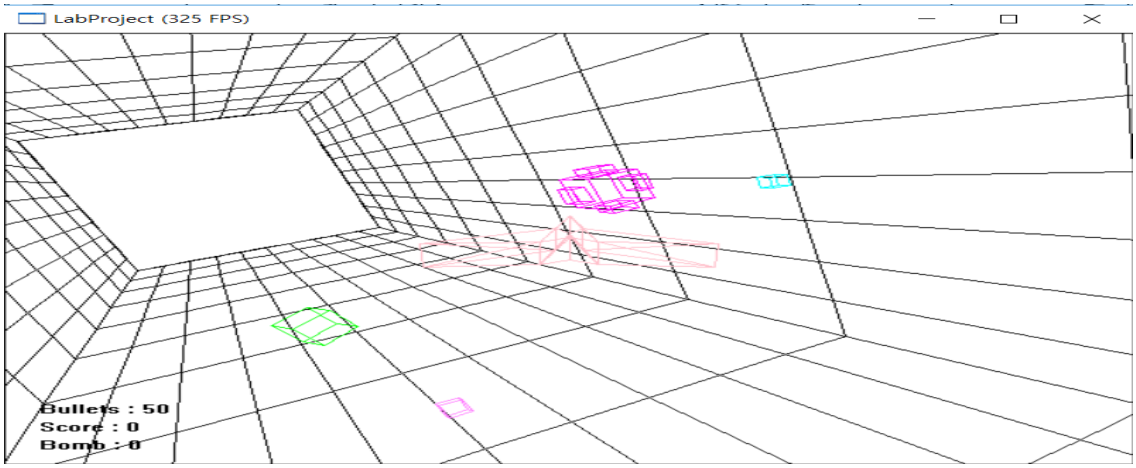
기본 오브젝트는 10점, 빨강 오브젝트의 경우 50점과 아이템을 준다.

아이템으로는 점수획득이 되지 않는다.

보스

맵 반대편에 존재하며, 플레이어 z좌표가 300 이상일시 등장한다..

이에 따라 보스에 해당하는 공격 및 충돌처리 등 함수들은 보스가 등장할 때부터 호출된다.



보스가 생성되고 나서는 주기적으로 생성되던 오브젝트 함수는 불리지 않고, 보스는 등장하고 3초 후 공격을 시작하고, 1초마다 플레이어를 향해 총을 쏘며, 보스의 2번째 패턴으로 보스가 회전하며 2초간격으로 2마리의 오브젝트를 소환한다. 생성된 오브젝트들은 2초를 랜덤이동하다 플레이어를 쫓는다. 이 때 소환오브젝트 중에서 빨강 오브젝트는 생성되지 않는다.

보스의 생명은 50이다.

플레이어는 보스에게 부딪히거나 총을 맞을 경우 게임을 재시작하게 되고, 제거시 100점과 함께 앤드플래그를 활성화하여, 게임의 끝을 알린다.

플레이어

플레이어는 기본 파랑색으로 노랑색의 총알 50을 가진다.

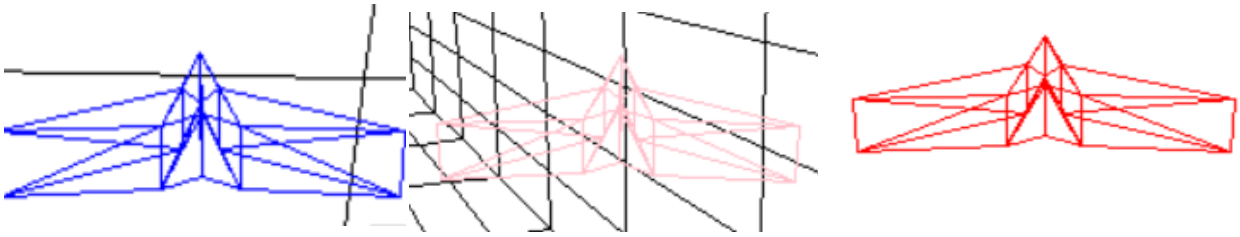
총알은 플레이의 Look벡터를 이용하여 바라보는 방향으로 나간다.

50발의 총알은 전부 없어져야만 다시 총알을 쏠 수 있으며, 총알이 없어지는 경우는 총알이 벽, 오브젝트에 부딪히거나, 플레이어로부터 거리가 100m보다 멀어 질 때이다.

적이 뒤에 있을 경우 (플레이어의 z 좌표보다 작을경우) 분홍색으로 바뀐다.

벽에 충돌했을 때는 빨간색으로 바뀐다. 적이 뒤에 있고 벽에 부딪혀도 빨강색이다.

살아 있는 오브젝트의 수가 99개가 될 시 게임 재시작이 된다.



기존 함수의 변경

Move함수

2개이상의 반대가 아닌 키 입력시 이동속도가 2배가 되는 경우가 있어 count 변수를 두어 들어왔을 때의 여러 키가 눌렸는 지 확인 후 반대키가 아닐 경우 속도 조정

Rotate함수

내부에서 자동적으로 라디안으로 바꿔주게 되어있는데 이미 라디안으로 값이 들어올 시 4번째 인자를 설정해주어 이 값이 이미 라디안일 경우 해당 연산없이 넘어갈 수 있게

조정

ProcessInput()

키보드 마우스 입력 및 피킹작업 처리

Animate()

충돌처리, 플레이어 배후의 적, 보스(플레이어와 거리 비교, 공격과 패턴변환), 총알 제거(벡터의 길이 비교), 피킹 해제(피킹 오브젝트가 폭발했는지 검사) 등 실시간으로 계산이 필요한 함수 처리 와 모든 오브젝트의 Animate호출

Render()

모든 오브젝트들은 월드변환부터 스크린변환까지의 과정을 거친 후 게임세상에 그려진다.

좌측 하단의 점수, 아이템 개수 등 표기

급격한 프레임저하를 막기 위하여 총알이나 보스, 오브젝트 등의 animate, render 등에서 플래그를 두어 예외처리를 하였다.