오델로

3D 오델로 영상 링크
https://youtu.be/JhOYmCS1E08

C++ 깃허브 링크
https://github.com/hoya1215/Cpp-AI-Othello-Game.git

DirectX12 엔진 링크
https://github.com/hoya1215/DirectX12-Engine.git

Table of Contents

목차

2D 오델로

구현 설명

오델로는 둘 수 있는 곳에 돌을 두어서 상하좌우 , 대각선으로 돌과 돌 사이에 있는 다른 돌을 전부 자기의 돌로 바꿔서 제일 많은 돌을 바꾼 사람이 승리하는 게임입니다.

초기 설정

1. 게임 판 사이즈 설정

게임 판의 가로 , 세로의 크기를 숫자를 입력하여 지정할 수 있습니다.

2. 게임 모드 설정

기존 오델로 모드와 게임 판의 랜덤한 위치에 일정 개수만큼의 벽을 설치하여 게임의 재미를 더 향상시킨 벽 모드를 선택할 수 있습니다.

3. 게임 난이도 설정

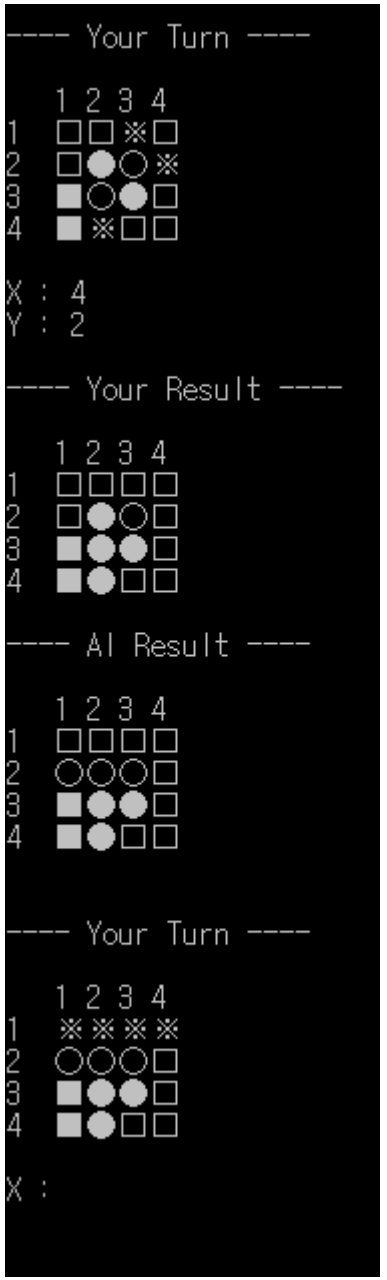
이지 , 노멀 , 하드 모드를 선택할 수 있습니다.
각 난이도에 따라 AI 가 돌을 두는 방식이 달라집니다.

2D 오델로

구현 설명

기호 설명

□ : 게임 판 기본 블록
● : 플레이어 돌
○ : AI 돌
■ : 벽 모드에서만 나오는 랜덤 벽
※ : 플레이어가 둘 수 있는 곳



게임 규칙

플레이어가 돌을 두기 전에 게임 판에서 돌을 둘 수 있는 곳을 특정 기호로 보여줍니다.

플레이어는 X (세로) , Y (가로) 좌표를 입력하여 돌을 둘 수 있습니다.
(둘 수 없는곳을 입력하면 재입력)

플레이어가 돌을 두면 돌을 두었을 때 바뀐 판의 모습이 먼저 보여지고
다음에 AI가 두었을 때 바뀐 판의 모습을 보여준 다음에
다시 돌을 둘 수 있는 곳이 표시된 판을 보여줍니다.

플레이어나 AI 가 각 자기 차례일때 돌을 둘 수 있는 곳이 하나도 없으면
다시 상대 턴으로 넘어갑니다.

서로 돌을 둘 수 없는 상황이거나 판이 모두 돌로 가득찬다면
각 플레이어들의 돌 개수를 보여주고 승자를 보여주면서 게임이 종료됩니다.

2D 오델로

AI 알고리즘

-이지 모드

AI 가 돌을 둘 수 있는 곳 중에 한 곳에 랜덤하게 둡니다.

-노멀 모드

AI 가 돌을 둘 수 있는 곳 중에 두었을 때 가장 많이 뒤집을 수 있는 곳에 돌을 둡니다.

-하드 모드

AI 가 (현재 돌을 두었을 때 가장 많이 뒤집을 수 있는 개수 - 그 위치에 두고 난 뒤에 플레이어가 두었을 때 뒤집을 수 있는 돌의 최대 개수) 의 기대값이 가장 높은 곳에 돌을 둡니다.

3D 오델로

구현 설명

- 2D 와의 공통점
알고리즘 방식 동일

- 2D 와의 차이점
2D 일때는 상하좌우 대각선 총 8번만 검사하면 되지만 3D 는 총 26번 검사를 해야 합니다.

4 x 4 x 4 만 되어도 대략 64 * 26 번 시행을 해야 하고 바꿔주려면 더 시행을 해야 합니다.

- GPU 에서 Compute Shader 로 최적화 하기
-> 크래시 발생
발생하는 이유를 찾아보니 if 문으로 검사하는 조건이 많았고 shader 에서는 if 문의 실행이 느리기 때문
실제로 if 문을 뺀 코드는 실행이 잘 되었습니다.

Player 의 실행 , AI 의 실행 을 매 프레임 검사하지 않고 바껴야 하는 상황에서 딱 한번만 검사할 수 있게 설정했습니다.

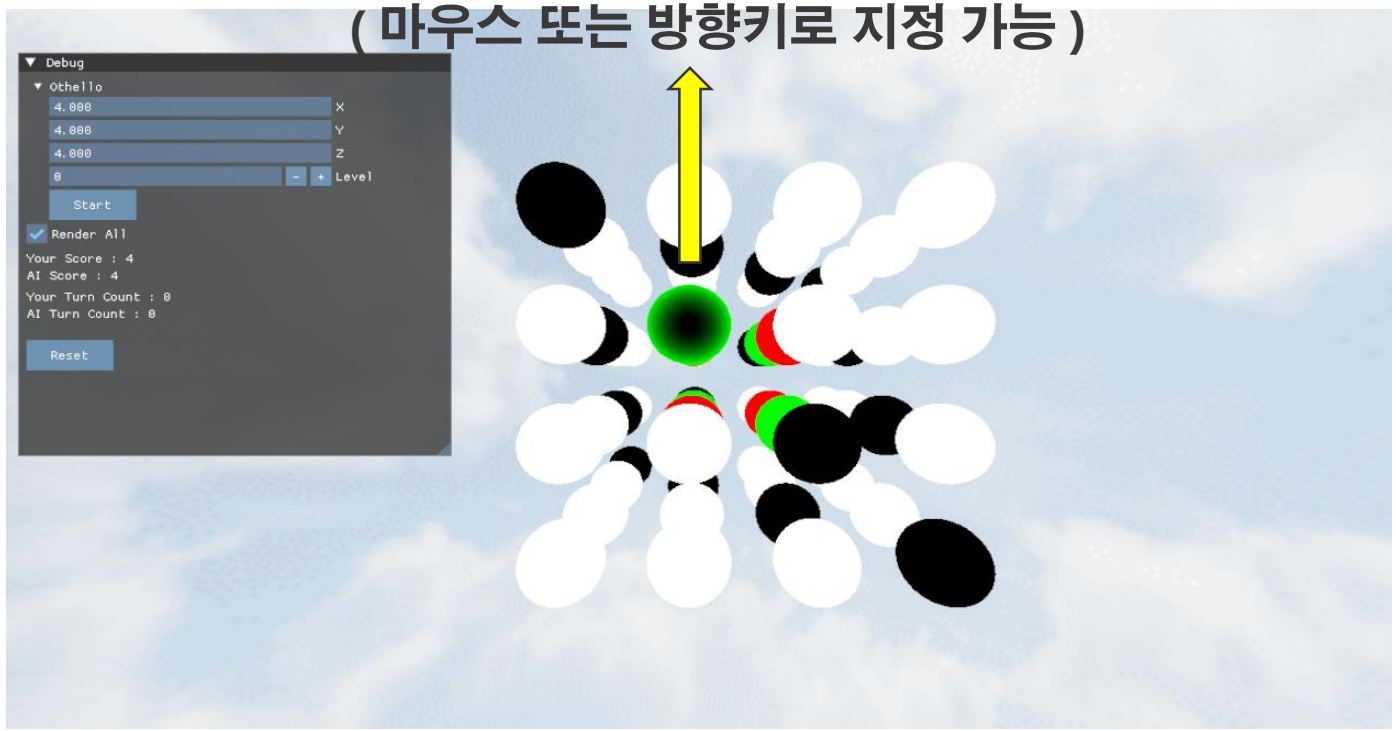
- 결과
8 x 8 x 8 이여도 로직 실행 시 프레임 드랍 일어나지 않음

3D 오델로

구현 설명

흰색 : 기본 물체 색 (보이기 On / Off 가능)
빨간색 : 플레이어 색
초록색 : AI 색
검은색 : 현재 놓을 수 있는 위치 색

현재 선택된 곳 표시
(마우스 또는 방향키로 지정 가능)



3D 오델로

개선할 사항

- **물체의 시인성 개선**

각 물체의 위치를 잘 볼 수 있도록 개선

- **시뮬레이션 적용**

플레이어의 Play , AI 의 Play 가 한 프레임에 모두 적용되어서 나타나므로 어떻게 바뀌는지 알아보기가 힘듭니다.

각 물체가 변하는 단계를 Timer 를 통해 제공