|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **작성자** | **2019180049 윤우영**  **2019180046 김강휘**  **2016180015 김지호** | **팀명** | Survil |
| **주차** | **1** | **기간** | **2022.01.02~2022.01.08** | **지도교수** | **정 내 훈**(서명) |
| **이번주 한일** | * **윤우영: Cuda를 사용하여 2차원 배열에서 맵 정보 제작** * **김강휘: 마우스 클릭을 통한 Citizen의 이동 구현** * **김지호: 2차원 배열의 맵 높이 차이 계산** | | | | |

**<상세 수행내용>**

윤우영:

처음에는 랜덤으로 언덕의 높이, 반지름, 위치를 정하여 언덕 정보 배열에 넣고, 전체 맵 (40000\*40000) 2차원 배열을 만들어 각 블록의 높이 정보를 Cuda를 사용하여 입력했다. RTX3070TI기준 약 9 초가 걸렸다. 그 후, 최대한 클라이언트에 적은 양의 데이터를 보내기 위하여 플레이어의 시야 안에 들어오는 블록(1000\*1000) 만큼만 전체 맵 (40000\*40000)에서 복사하는 방식으로 했지만, 굳이 처음에 전체 맵을 만들지 않고 플레이어에게 맵 정보를 전송해야 할 때, 플레이어의 시야 안에 들어오는 블록(1000\*1000)을 언덕 정보 배열을 참조하여, 그때 그때 생성하는 방식으로 하는 것이 나을 것으로 판단하였다. 그래서 다음과 같은 방식으로 구현했다.

랜덤으로 언덕의 높이, 반지름, 위치를 정하고 전체 맵 (40000\*40000) 중 플레이어의 시야 안에 들어오는 블록(1000\*1000)만을 update하였음(Cuda 사용). 플레이어의 카메라 위치를 임의로 만들어, 카메라의 위치가 바뀜에 따라 시야 안에 들어오는 블록(1000\*1000)이 달라지는 것을 확인하였고, 지형의 이동에 따라서 달라지는 것도 확인했다.

아래는 플레이어에게 보낼 1000\*1000 배열을 업데이트 할 시, 걸리는 시간이다. 평균 0.09초 정도 걸린다. (RTX3070TI, VS release mode)

Player Sight Update Time : 0.087 Seconds

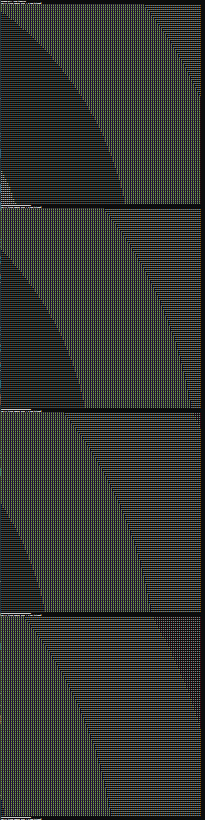
Player Sight Update Time : 0.089 Seconds

Player Sight Update Time : 0.089 Seconds

Player Sight Update Time : 0.083 Seconds

아래는 cmd에 출력된 100\*100(1000으로 하면 너무 커서 캡쳐 하기 힘들어서 줄임) 배열이다.

지형의 위치가 바뀐 것을 확인할 수 있다.



김강휘:

언리얼 에디터에 Citizen 1개 미리 배치 후 왼쪽 마우스 클릭을 통해 이동을 구현 목표

MyPlayerController라는 컨트롤러 클래스를 사용해 구현

좌클릭으로 클릭하면 해당 위치로 이동

클라이언트에서의 먼저 구현

Test\_1 결과

텍스트, 모래언덕이(가) 표시된 사진

자동 생성된 설명 

클릭 시 마우스 위치로 이동

고쳐야 할 부분

- 현재 순간이동 형식으로 이동하여 서서히 이동하도록 구현 (서버에서 구현 예정)

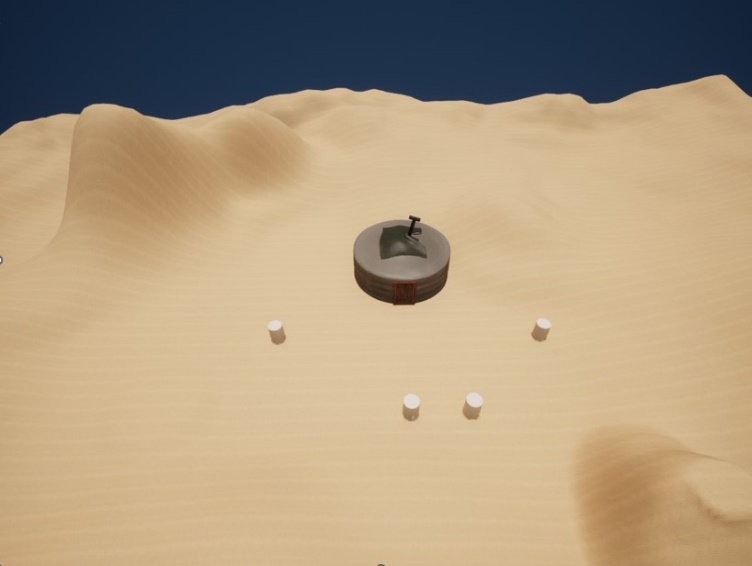
- 현재 미리 지정된 citizen만 이동 좌클릭으로 대상 선택 후 우클릭으로 이동해야 함

- 서버 미 연동

-

Test\_2

클릭 시 마우스 위치로 이동

고쳐야 할 부분

- 현재 순간이동 형식으로 이동하여 서서히 이동하도록 구현 (서버에서 구현 예정)

- 모든 Actor가 마우스 클릭 위치로 이동(citizen, 건물 포함)

- 마우스 클릭 시 Actor 구분을 할 수 있어야 함 (tag를 이용한 구분)

- 서버 미 연동

김지호:

2차원 배열의 값을 읽어 블록과 근접한 블록들의 높이 차이를 계산한다.

이미 계산한 블록들의 높이 차는 다시 계산하지 않는다.

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** |  | **해결 방안** |  |
| **다음 주차** | **2** | **다음 기간** | **2022.01.09~2022.01.16** |
| **다음주 할 일** | 윤우영:   1. 풍향과 풍속 값을 주어, 맵이 이동하는 것을 구현해야 한다.   김강휘:   1. Citizen의 이동을 서버에서 구현   김지호:  Unreal5에서 CUDA 적용하기. | | |
| **지도교수**  **Comment** |  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **작성자** | **2019180049 윤우영**  **2019180046 김강휘**  **2016180015 김지호** | **팀명** | Survil |
| **주차** | **2** | **기간** | **2022.01.09~2022.01.15** | **지도교수** | **정 내 훈**(서명) |
| **이번주 한일** | * **윤우영: 풍향과 풍속 값을 주어 언덕 이동, 삭제, 생성** * **김강휘: citizen의 이동을 서버에서 구현** * **김지호: Unreal5에서 CUDA 적용하기** | | | | |

**<상세 수행내용>**

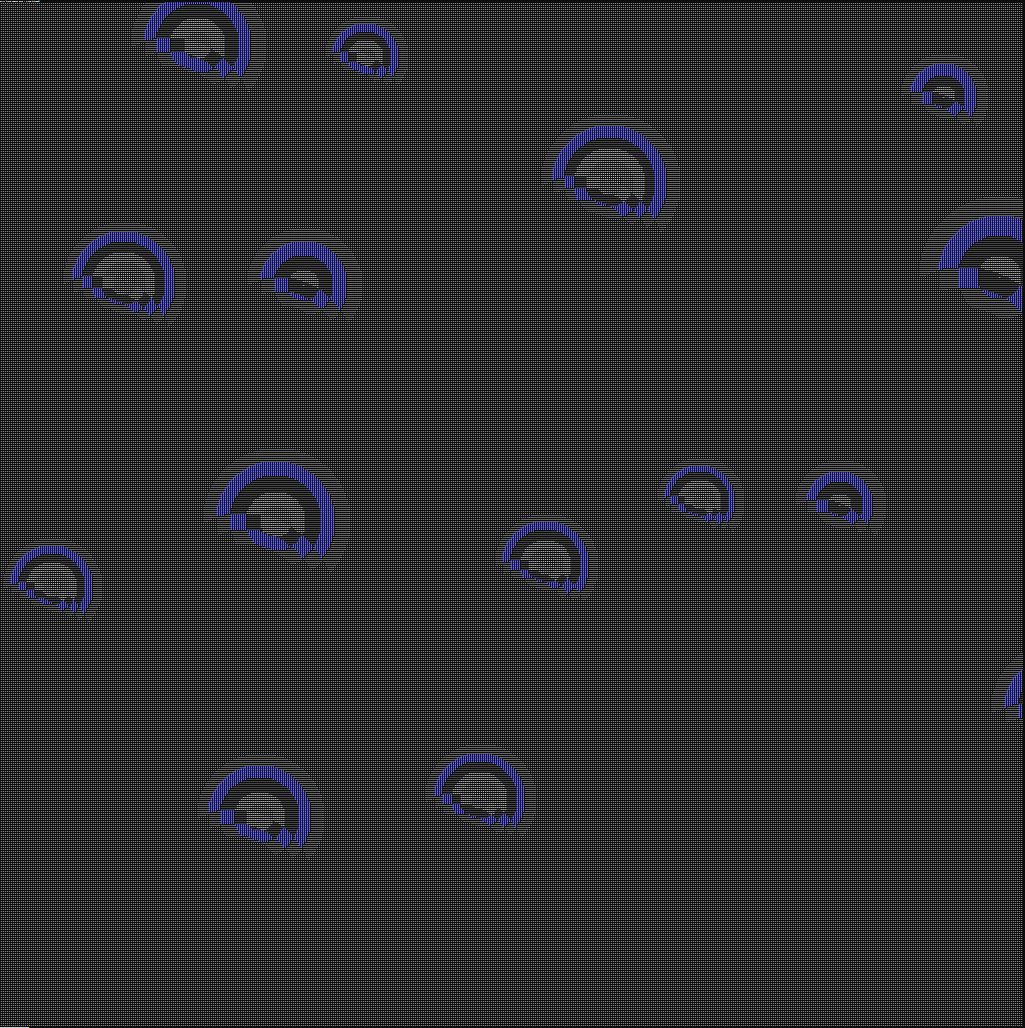
윤우영:

풍향과 풍속 값을 주어서 맵 전체가 이동할 수 있게끔 하고, 바람에 의해 언덕이 맵 밖으로 나가게 되면, 언덕 정보 배열에서 언덕 정보를 지워주고, 바람이 불어오는 방향에서 새로운 언덕을 생성하여, 처음에 생성되었던 언덕의 개수가 유지되게끔 했다.

아래 사진은 전체 맵 512\*512에서 풍속과 풍향 값을 주어, 언덕이 풍향에 맞게 이동하는 모습이다. 풍향에 따라 언덕이 바람에 의해 깎인 정도가 달라지고, 아래 사진은 최고 풍속(50)일 때의 모습이다. 풍속이 0이면 언덕의 모양은 완전한 원 모양이고, 최대 풍속에 가까워질수록 언덕이 깎이는 정도가 심해진다. 풍향에 업데이트마다 10도를 더해주어, 언덕이 바라보는 방향, 이동하는 방향이 약간씩 시계방향으로 돌아가는 것도 확인할 수 있다.

자연이(가) 표시된 사진

자동 생성된 설명



자연이(가) 표시된 사진

자동 생성된 설명



아래 사진은 40000 \* 40000 전체 맵 중 1000 \* 1000의 맵을 업데이트 하는데 걸리는 시간이다. (풍향, 풍속 적용)

텍스트이(가) 표시된 사진

자동 생성된 설명

김강휘:

Citizen 연동 시작

보내야할 데이터와 받아야 할 데이터 정리

* 초기 Citizen의 개수, Citizen에 대한 정보 (name, location, rotate)
* 마우스 입력을 통한 Citizen의 정보와 다른 Citizen의 정보

MyPlayerController와 Server\_Testing 클래스를 연동하여 MouseInput에 따른 정보 전송

Map을 이용해 필요한 정보를 저장하고 Struct Factor\_location\_rotation의 정보를 전송하여

현재 위치와 이동해야 하는 위치를 서버에서 작업

Test1 :

Citizen의 이름과 현재 위치 전송

오류로 인한 실행 불가능

* FString으로 이름을 불러오는데 언리얼은 char형식에는 정보를 이동하지 못함

해결방안

* FString는 유니코드 형식이라 TCHAR를 이용해야 한다.

Test2 :

Citizen의 이름과 현재 위치 전송

Server\_test에서 현재 월드에 있는 모든 Citizen을 찾아낸다.

먼저 현재 월드에 있는 모든 Citizen의 개수를 전송 후 이후 그 수만 큼 반복 전송

텍스트이(가) 표시된 사진

자동 생성된 설명

현재는 4개의 Citizen이 반복됨

Test3 :

MouseInput에서 입력받아 도착 location을 Server에 보내주는 작업

두 Class간의 함수 호출이 되지 않는 문제

오류를 찾아봐도 해결되지 않음. 인터넷을 찾아봐도 해결되지 않음

* 이 과정에서 많은 시간 소모
* world에서 server Actor를 찾아 ref를 이용하여 Cast<AServer\_testing>로 재 캐스팅 해줘야 함

Test4

모든 Citizen이 목표 지역값에 따라 이동



각 Citizen이 목표 지역에 따라 이동

문제점:

아직 모든 플레이어가 공통된 Citizen을 조작가능

* 차후 자신의 Citizen만 조작 가능하게 구현

Citizen의 충돌 처리가 되지 않음

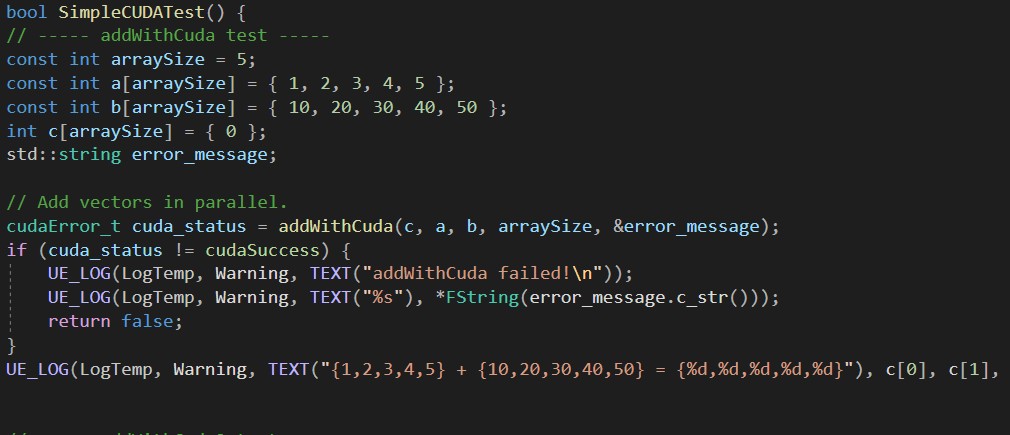
* 현재 일직선으로 이동함 건물을 뚫고 지나감

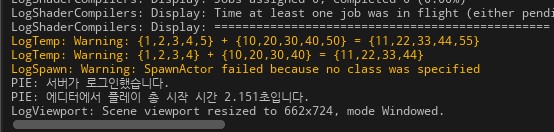
김지호:

CUDA를 언리얼에서 사용하기 위해 CUDA 정적 라이브러리를 생성.

CUDA라이브러리를 적용시키기 위해서 언리얼의 .build.cs 파일 변경.

CUDA 함수가 언리얼에서 실행되는 것을 확인했다.





문제점:

필요한 CUDA 함수가 있을 때마다 CUDA 라이브러리에 추가해주어야 한다.

테스트용으로 새 프로젝트에서 했을 때는 빌드가 잘 됐지만, 현재 프로젝트에서 적용해 보려고 했을 때는 빌드가 안 된다.

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** |  | **해결 방안** |  |
| **다음 주차** | **3** | **다음 기간** | **2022.01.16~2022.01.22** |
| **다음주 할 일** | 윤우영:   1. 마을의 위치를 랜덤으로 생성하고, 그 부분은 언덕이 생성되지 않고, 지형이 이동하더라도 높이가 0으로 되게끔 해야 한다. (절벽이 생기면 안됨: 블록 1칸에 높이의 차이가 1을 넘으면 안됨) 2. 언덕의 모양을 원 말고도 여러 종류를 만들어야 함.   김강휘:  서버에서 자원 위치 랜덤 구현 후 클라이언트에게 전송  Citizen을 클릭하여 자원에 이동시키면 자원을 채취 후 창고에 넣는 AI 구현  김지호:  현재 프로젝트에서 빌드가 안되는 이유를 찾고 해결.  CUDA로 블록들 생성하기. | | |
| **지도교수**  **Comment** |  | | |