

Holo-City

- Nui를 이용한 건축 시뮬레이션 - 프로젝트 완료 보고서

- 제정일자 : 2018년 11월 20일
- 문서버전 : Ver 2.0
- 팀명 : C-Nergy

문 서 승 인 정 보

프로젝트 명	Holo-City
TASK 명	Nui를 이용한 건축 시뮬레이션
문 서 명	프로젝트 완료보고서
발행 년 월일	2018년 11월 20일

구 분	성 명	서 명	일 자
Team Leader	조덕진		2018.11.26
Team members	김영서		2018.11.26
Team members	이태경		2018.11.26
Team members	윤혜진		2018.09.19
Team members	김도현		2018.09.19
Team Mentor	문상환		2018.11.26

문서이력정보

[illegible]

목차

1. 프로젝트 개요
 - A. 프로젝트 명칭
 - B. 프로젝트 주제
 - C. 프로젝트 가치
2. 프로젝트 진행보고
 - A. 프로젝트 개발 분야
 - B. 프로젝트 진행 일정
 - C. 수행업무 및 담당자
 - D. 개발 도구
 - E. 프로젝트 개발 방향
3. 프로젝트 개발 일정
 - A. 팀 개발 일정
 - B. 개인 개발 일정
 - C. 회의록
 - D. 개인 개발일지
 - i. 김영서 개발일지
 - ii. 조덕진 개발일지
 - iii. 이태경 개발일지
4. 프로젝트 개발 내용
 - A. 프로젝트 배경 지식/기술/알고리즘
 - B. 프로젝트 상세 개발 내용
5. 사용자 매뉴얼
6. 프로젝트 마무리
 - A. 기대효과
 - B. 문제점
 - C. 개선방안
 - D. 참고문헌 및 논문
 - E. 참고사이트
 - F. 팀원 별 소감

1. 프로젝트 배경

A. 프로젝트 명칭

- i. 명칭 : Holo-City
- ii. 의미 : '완전함' 혹은 '전체' 라는 뜻의 'Holo'와 'City'의 결합을 의미

B. 프로젝트 주제

- i. 분야별 개발
 - 홀로그램 / VR : 건설/건축 시뮬레이션 홀로그램 영상 제공 및 동시 출력
 - Unity : 지형/건물/도로 등등 건축 기본 소재 제공 및 사용자 인터페이스 UI 제공
 - Leap Motion : 손 동작을 통한 프로그램 제어/관리
 - 서버 / 데이터 : Asset Bundles 활용한 건축 시뮬레이터 장면 및 건축 소재 저장

C. 프로젝트 가치

- 목적
 - VR과 Leap Motion을 결합한 건축 시뮬레이션을 구축, 사용자가 편하게 건축 디자인을 할 수 있게 도움을 주고, 홀로그램을 활용하여 사용자가 해당 시뮬레이션을 입체감 있게 볼 수 있게 한다.
 - 여러 가지 상황 부여 시뮬레이션을 통해 해당 상황에 따른 대처 방법을 제시 한다.
- 이점
 - 기존 GUI 건축 설계한 틀을 깬 NUI 환경을 지원, 보다 생동감 있는 건축 설계를 보장 한다.
 - 사용자가 건축 설계 시 실시간 홀로그램을 지원한 진행상황을 볼 수 있다.

- 필요성, 수행이유
 - 기존의 건축 설계 방식은 GUI 환경에서 설계 작업을 하지만 본 프로젝트에서는 NUI 환경을 제공하여 사용자가 보다 편하고 원하는 환경에서 건축 설계가 가능하다.
 - 기존 건축 시뮬레이션 조형도 설계에 시간을 들일 필요 없이 홀로그램을 통해서 입체감 있는 건축 시뮬레이션 조형도를 실시간 출력이 가능하다.

2. 프로젝트 진행 보고

A. 프로젝트 개발 분야

(가) Unity 3D를 이용한 건축시뮬레이션 구현

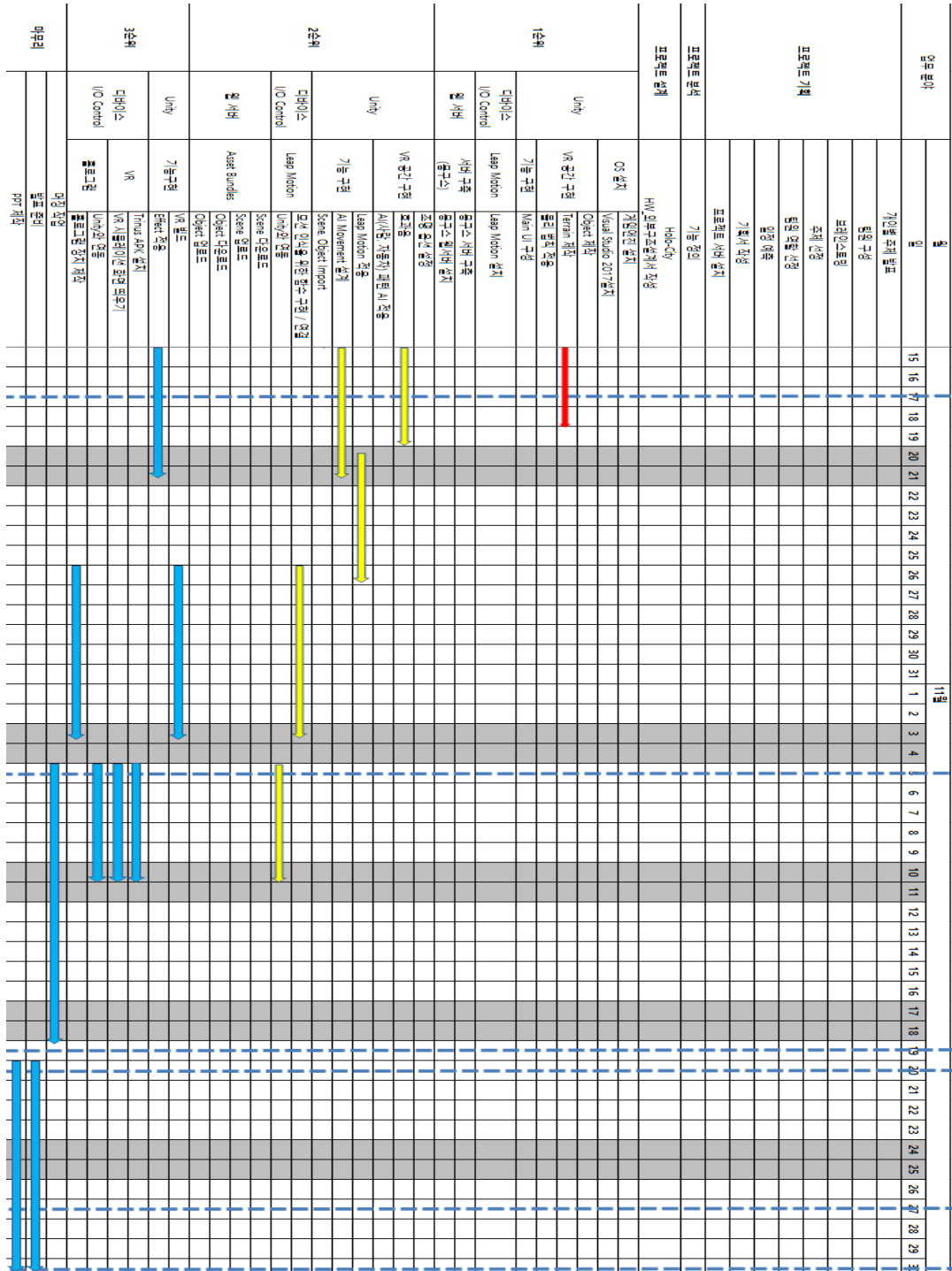
(나) Leap Motion 기능을 지원한 NUI 환경 조성

(다) 홀로그램을 사용한 입체적인 화면 출력

(라) TrinusVR을 사용한 생동감 있는 연출


(마) 설치 및 설계가 가능한 UI모드, 동적인 상황 변화를 지원하는 AI모드 기능 구현

[illegible]



C. 수행업무 및 담당자

구분	사진	이름	역할 설명
Team Leader		조덕진	① 일정관리/기획서관리 ② Leap Motion 개발 ③ 동적 환경요소 추가 ④ 상황 별 행동 패턴 AI 제작 ⑤ 홀로그램 제작 및 구현 ⑥ 영상 실시간 동시 처리 구현 [VR / PC / 홀로그램 간 통신]
Team Member		김도현	① 서버 / Asset Bundles 구현 ② Leap Motion 개발 ③ 동적 환경요소 추가 ④ 상황 별 행동 패턴 AI 제작
		김영서	① Save/Load 기능구현 ② Leap Motion 개발 ③ 동적 환경요소 추가 ④ 상황 별 행동 패턴 AI 제작 ⑤ 홀로그램 제작 및 구현 ⑥ 영상 실시간 동시 처리 구현 [VR / PC / 홀로그램 간 통신]
		윤혜진	① 데이터 수집 관리 ② Object 제작 및 디자인 설계

		이태경	<ul style="list-style-type: none"> ① 전체 UI 제작 ② AI 네비게이션 기능 ③ AI 인공지능 제작 ④ 상황에 맞는 퍼지로직 기능 구현 ⑤ 보고서 및 프로젝트 문서 작성
--	---	-----	--

D. 개발 도구

I. 사용 기술

#	이름	사용처
1	C#	Unity를 통한 시뮬레이션 클라이언트 구현
2	Trinus VR	PC & mobile 실시간 이종의 통신
3	Leap Motion	손을 인식 하여 I/O을 Control

II. 개발 S/W

#	이름	사용처
1	Visual Studio 2017	프로젝트 표준 개발 도구
2	Unity	
3	Window 8 / 10	PC Application 동작 환경
4	Leap Motion SDK	Leap Motion 개발 Tool, 개발 환경 조성

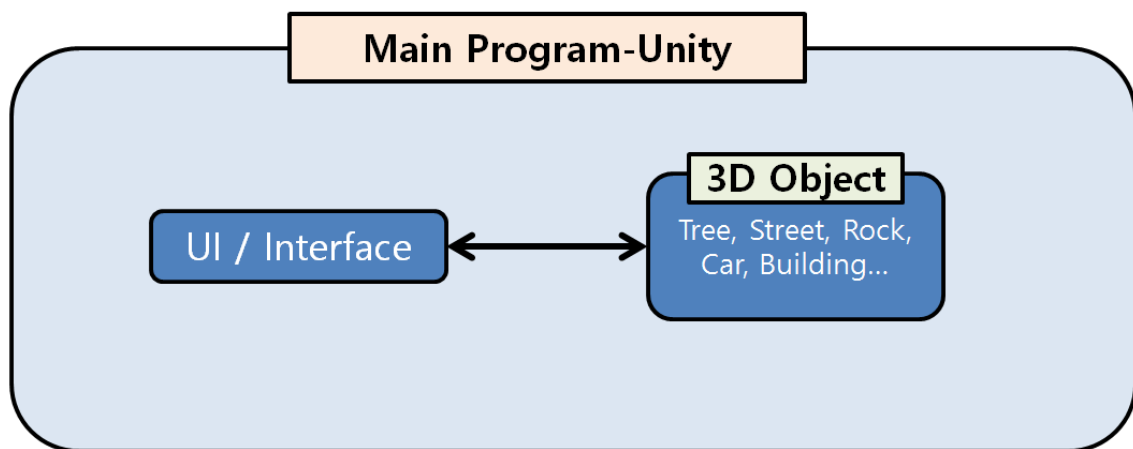
III. 장비

#	이름	사용처
1	VR 기기	시뮬레이션 화면을 VR로 출력
2	PC	Windows PC: 5대 개인 개발 및 테스트 환경 제공
3	홀로그램 장치	시뮬레이션 화면을 홀로그램으로 띄움
4	LeapMotion	적외선 센서를 이용한 손 Motion 인식 영상 처리를 통한 3D Hand Model 구현

E. 프로젝트 개발 방향

아키텍처 1단계

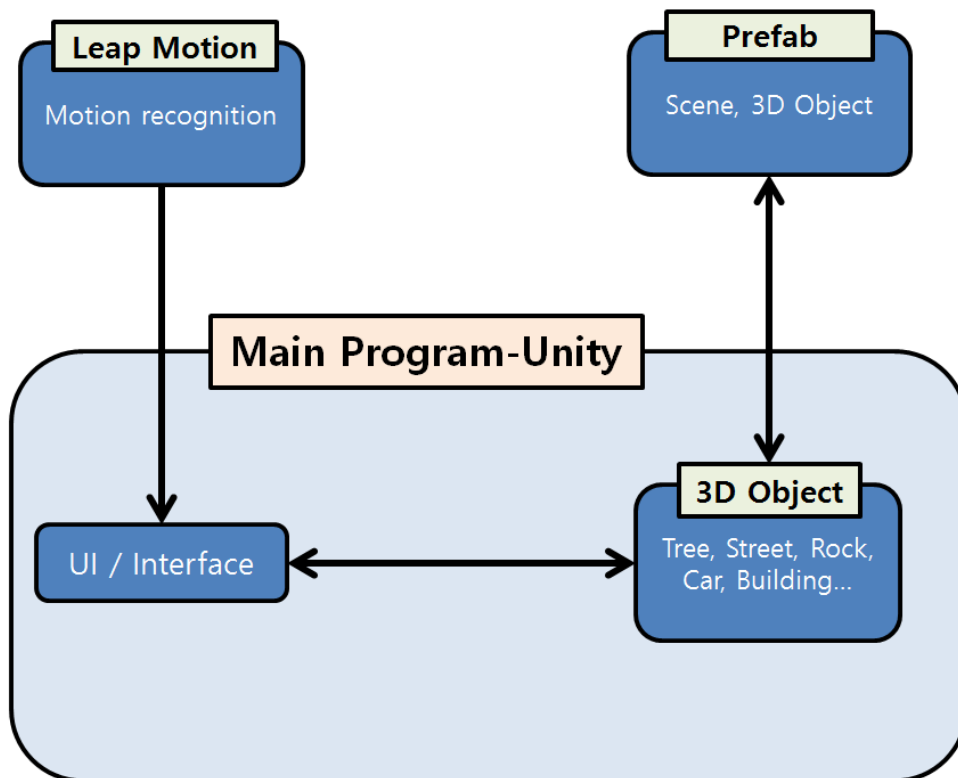
- 1단계 주요 주제
- 전체 프로젝트 달성을 위한 기본적인 기능들을 구현



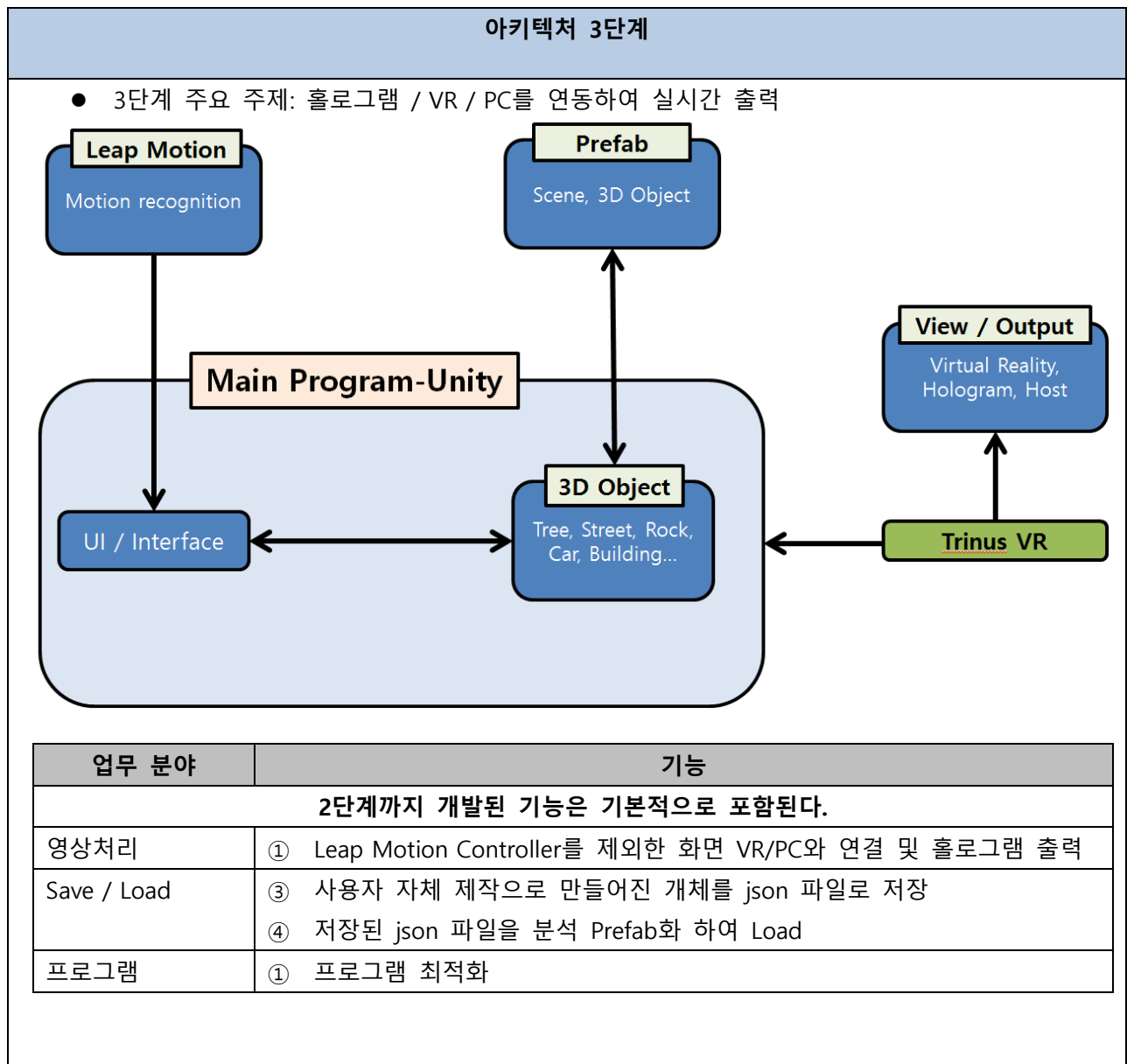
업무 분야	기능
영상 처리	① 평면(2D) or 입체(4분면) 구조의 홀로그램 출력 화면 결정을 위한 토의
Leap Motion	① Leap Motion 툴 설치 및 구현 ② 손 동작을 통한 프로그램 제어 테스트
Unity	① 기본적인 UI, 카테고리 제작 ② 사용자 인터페이스를 통한 건물 배치 기능 구현

아키텍처 2단계

- 2단계 주요 주제
- 기존 Mouse Controller를 Leap Motion Controller로 대체
- 서버에 Asset Bundles 기술을 도입 서버에 데이터 베이스 기능을 부여



업무 분야	기능
1단계까지 개발된 기능은 기본적으로 포함된다.	
영상 처리	① PC와 VR의 연동 ② 입체(4분면) 구조의 홀로그램 구현 및 제작
Leap Motion	① Unity 연동 ② 기존 Mouse Controller를 Leap Motion Controller로 대체
Unity	① 시뮬레이션 동적 모드(사람/자동차 등등 개체들의 다양한 움직임) 추가 ② 시뮬레이션 상황 모드(날씨/시간/재난 등등 다양한 상황 제어 기능) 추가
Save / Load	① 사용자 자체 제작으로 만들어진 개체를 json 파일로 저장 ② 저장된 json 파일을 분석 Prefab화 하여 Load



3. 프로젝트 개발 일정

A. 팀 개발 일정

1 주차 : 2018 년 9 월 (9 월 27 일 ~ 9 월 28 일)		
담당자	업무이름	업무 내용
조덕진	몽구스 웹 서버 설치	웹 서버 구축을 위한 몽구스 웹 서버 프로그램 설치
김도현	Leap Motion 설치	Leap Motion 작동을 위한 SDK 설치
김영서	개발환경 구축	개발에 필요한 작업환경 및 프로그램 설치, 구축
윤혜진	Object 제작	시뮬레이션 내의 배치할 요소 수집
이태경	메인 UI 구성	메인 UI 구조 설계 및 배치
2 주차 : 2018 년 10 월 (10 월 1 일 ~ 10 월 5 일)		
담당자	업무이름	업무 내용
조덕진	몽구스 서버 구축	웹 서버와 Asset Bundles 파일 경로 동기화
김도현	Asset Bundle 에 Scene 업로드/ 다운로드 작업확인	구축된 Asset Bundles 를 이용하여 시뮬레이션 Scene 정보를 저장
김영서	물리법칙 적용	샘플 Object 를 이용하여 물리 충돌 및 중력에 대한 Default 값 설정
윤혜진	Object 제작	시뮬레이션 내의 배치할 요소 추가 제작
이태경	메인 UI 구성	각 카테고리별 세부항목 UI 제작
3 주차 : 2018 년 10 월 (10 월 8 일 ~ 10 월 12 일)		
담당자	업무이름	업무 내용
조덕진	Scene, Object Import	Asset Bundles 를 통한 Scene, Object Import
김도현		
김영서	조명 옵션 설정	시간 변화에 따른 조도 작성
윤혜진	Terrain 제작	공간에 적절한 샘플 지형 제작
이태경	동적 AI 제작	상황에 따른 동적 요소(패턴 인식 AI) 설계
4 주차 : 2018 년 10 월 (10 월 15 일 ~ 10 월 19 일)		
담당자	업무이름	업무 내용

조덕진	AI movement 설계	특정 상황에 따른 동적 움직임 구현
김도현	Effect 적용	UI 와 이벤트를 통한 Effect 적용
김영서	효과음	UI 이벤트 효과음, 상황 부여에 따른 효과음
윤혜진	Terrain 제작	공간에 적절한 샘플 지형 제작
이태경	AI movement 설계	특정 상황에 따른 동적 움직임 구현
5 주차 : 2018 년 10 월 (10 월 22 일 ~ 10 월 26 일)		
담당자	업무이름	업무 내용
조덕진	특수 상황 AI 조건 추가	특정 상황에 따른 AI 의 시뮬레이션 구현
김도현	Leap Motion 적용	Leap Motion 입력 데이터 전송 기능을 확인
김영서		
윤혜진	홀로그램 장치 제작	홀로그램을 구현할 수 있는 장치 제작
이태경	특수 상황 AI 조건 추가	특정 상황에 따른 AI 의 시뮬레이션 구현
6 주차 : 2018 년 11 월 (10 월 26 일 ~ 11 월 2 일)		
담당자	업무이름	업무 내용
조덕진	VR 빌드	프로젝트를 VR 콘텐츠 기반으로 빌드 기능을 확인, VR 기기와 Unity 를 연동하여 시뮬레이션 테스트 및 오류 확인 수정
김도현	모션 인식을 위한 함수 구현 / 연결	Leap 을 이용해 손 위치 좌표 및 손 동작을 인식하여 사용자가 원하는 지시를 분간 지을 수 있게 한다.
김영서		
윤혜진	홀로그램 장치 제작	시험 제작한 장치를 작동해보며 오류 수정 작업
이태경		
7 주차 : 2018 년 11 월 (11 월 5 일 ~ 11 월 9 일)		
담당자	업무이름	업무 내용
조덕진	홀로그램, VR, Leap Motion, Unity 연동	팀원들의 업무결과(Leap Motion, VR, 홀로그램)를 하나의 기능으로 종합, 연결하여 기능을 TEST.
김도현	Trinus APK 설치/ Unity 와 연동	Trinus APK 설치, Unity 를 이용한 완성된 프로그램을 Trinus VR 과 실시간 연동 확인
김영서	Untiy 와 연동	Unity 와 연동 사용자의 손동작을 인식하여

		Unity 에서 표현할 수 있도록 연동
윤희진	홀로그램, Unity 연동	Unity 건축 시뮬레이션 장면을 실시간 홀로그램 영상으로 송출
이태경		
8 주차 : 2018 년 11 월 (11 월 12 일 ~ 11 월 19 일)		
담당자	업무이름	업무 내용
조덕진	최종 마무리작업	홀로그램, VR, LeapMotion 연동 최적화
김도현		
김영서		
윤희진		
이태경		

B. 개인 개발 일정

날짜		09 월 27 일
1 회차	조덕진	Asset Bundle 구조 파악 및 웹 서버 설치 및 경로 설정
	김도현	Leap Motion 작동 원리 및 구조 파악, 작동 테스트
	김영서	Leap Motion 기능별 예제 학습 및 작동 테스트
	윤혜진	Object Data 를 수집, 기본 예제 Object 제작시작
	이태경	메인 Unity 카테고리 배치 및 상세 카테고리 항목 구성
날짜		9 월 28 일
2 회차	조덕진	서버와 파일의 경로를 동기화, Sample Object 를 이용한 업로드, 다운로드 테스트 수행 Leap Motion Gesture 기능 코드 작성
	김도현	Leap Motion Gesture 기능 코드 작성
	김영서	Leap Motion 예제를 바탕으로 물리 기능 추가 구현 및 Gesture 코드 구상
	윤혜진	Object Data 를 수집, 기본 예제 Object 제작 및 예제 Terrain 디자인
	이태경	각 카테고리 창 마다 개별 Object 리스트 제작
날짜		10 월 1 일
3 회차	조덕진	손 모션을 이용한 Object 를 카메라 시점이동, 위치이동 기능 코드 구현 Object 의 크기를 모션으로 제어 및 조정하는 기능 구상
	김도현	Leap Motion 를 Main Controller 로써 대체 적용 마우스 클릭 대신 손 인식 클릭 코드 작성 및 구현
	김영서	Leap Motion Controller 의 기능 중에 Object 를 Grap 모션 구현 및 코드 작성
	윤혜진	Object Shader 예제 학습 및 환경에 따른 Sample Terrain 디자인 제작 수행
	이태경	Object 리스트 제작 구현 완료 인벤토리창 제작 및 디자인 완료 후 구현
날짜		10 월 02 일
4 회차	조덕진	Object 의 크기 제어 및 조정 코드 구현 Sample UI 를 이용하여 이를 카메라에 3D 오브젝트화 수행
	김도현	손 인식을 이용한 클릭 이벤트 적용 코드 구현 Sample UI 와 연동 후 테스트 진행 및 코드 정상 작동 확인

	김영서	Grap 모션을 오브젝트에 적용시켜 잡아 위치를 이동시키는 코드 작성 완료 Object 에 물리법칙을 적용하여 구현하였을 때의 오류 확인 및 개선방법 구상
	윤혜진	우선적으로 1 개의 Sample Terrain 제작 완료 Shader 의 원리 및 특정 예제 학습 후 테스트 완료
	이태경	Object 리스트 내에 속해 있는 Object 활성화 구현, 시뮬레이션 내에서 테스트 완료
날짜		10 월 04 일
5 회차	조덕진	Gesture 을 이용한 구현된 기능코드들 과 Unity UI 를 전부 병합하여 작동되는지 테스트진행 전체적 작동시 발생하는 오차범위 최적화
	김도현	카메라의 상하시점이동 기능 코드 작성 이에 동반되는 오류 수정/ 대처 방안 탐색 및 최적화
	김영서	Object 의 Rigidbody 의 IsKinematic 기능을 이용하여 물리법칙 적용 및 오류 사항 수정
	윤혜진	Shader 의 원리 학습 및 sample Object 제작 및 Effect 코드 구현
	이태경	Unity UI 에 On/Off 버튼을 추가하여 동적 시뮬레이션 요소를 실행시키는 기능을 구현 및 기능추가 세부 Object 별 AI 요소 및 동적 움직임 기능 알고리즘 구상 및 디자인
날짜		10 월 05 일
6 회차	조덕진	병합 작업에서 생긴 UI 와 Leap Motion 사이의 오류를 수정 및 최적화 UI 창 의 카테고리별 Sample Object 를 DataBase 에 삽입/ 생성 후 구현된 각 기능 코드 작동 테스트
	김도현	Camera 의 상하 각도 조절을 위한 Gesture 결정 및 제어 방법 결정 및 구현 발생하는 오차범위 및 오류사항 수정, 최적화
	김영서	Camera 의 상하 각도 조절을 위한 Gesture 결정 및 제어 방법 결정 및 구현 발생하는 오차범위 및 오류사항 수정, 최적화
	윤혜진	Shader 를 이용한 sample Object(Ocean) 제작 구현 2 번째 Sample Terrain 구상 및 디자인
	이태경	Unity UI 의 추가사항/개선사항 추가 기능 구현 AI 요소 및 동적 움직임 기능 알고리즘 구상 및 네비게이션 효과 기능 구상
날짜		10 월 08 일
7 회차	조덕진	2 순위 작업 중 시뮬레이션 내 특정 AI 행동에 대한 상황 구상(건물붕괴/지진)

	김도현	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(날씨 전환)
	김영서	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(Skybox 변환을 이용한 계절변화)
	윤혜진	수집한 Object Data 들과 제작한 1 차 Sample Terrain 을 인벤토리 UI 삽입 및 DataBase Object 리스트에 적용
	이태경	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(Navigation 을 이용한 사람, 자동차 이동)
날짜		10 월 10 일
8 회차	조덕진	Update 되고 작업이 완료된 Object 데이터들과 인벤토리 UI 를 추가 병합 작업 이에 해당하는 구현된 기능 테스트 및 오류 수정
	김도현	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구현(날씨 전환) 2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(태풍재난상황)
	김영서	미참석
	윤혜진	Unity 예제 학습 및 스크립트 공부
	이태경	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(Navigation 을 이용한 사람, 자동차 이동)
날짜		10 월 11 일
9 회차	조덕진	Object 들을 UI 인벤토리 DataBase 에 이미지 및 코드 추가 작업 완료 및 기능 테스트 필요한 3 가지 배경 Terrain 구상 및 제작 시작
	김도현	날씨 전환 작업을 Sample UI 를 제작 후 테스트 및 기능 시험/ 코드구현 완료 2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(태풍재난상황)
	김영서	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구현(Skybox 를 사용한 계절 전환) 2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상/ 알고리즘 구상(낮/밤 조도변화 상황)

	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구상 AI movement 의 해당 코드/ 알고리즘 구상(Navigation 을 이용한 사람, 자동차 이동)
날짜		10 월 12 일
10 회차	조덕진	Object 의 이동할 시에 Object 의 표면을 기준으로 Slide 되는 개선방안 결정 및 스크립트를 구현하기 위한 탐색 및 구상
	김도현	2 순위 작업 중 시뮬레이션 내 특정 AI 의 행동에 대한 상황 구현완료, AI movement 의 해당 코드/ 알고리즘 구현(태풍재난상황) 구현된 동적 상황요소들(날씨, 태풍, 지진, 붕괴 계절 등)을 하나로 통합하고 이에 UI 를 제작 후 적용시켜 기능테스트
	김영서	Object 을 배치할 시에 Object 의 인접 부분을 자동적으로 접촉되게 하는 개선방안 결정 및 스크립트를 구현하기 위한 탐색 및 구상
	윤희진	미참석
	이태경	AI movement 의 해당 스크립트/ 알고리즘 구현 (Navigation 을 이용한 사람과 자동차 이동을 구체적으로 구현 및 차별화)
날짜		10 월 15 일
11 회차	조덕진	Object 의 이동할 시에 Object 의 표면을 기준으로 Slide 되는 개선방안 결정 및 스크립트를 구현하기 위한 탐색 및 구상
	김도현	Object 리스트에 등록시킨 Object Data 들의 크기 표준화, Building Object 들에 Navi 기능을 위한 Gizmo 추가
	김영서	Object 을 배치할 시에 Object 의 인접 부분을 자동적으로 접촉되게 하는 개선방안 결정 및 스크립트를 구현하기 위한 탐색 및 구상
	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	AI movement 의 해당 스크립트/ 알고리즘 구현 (Navigation 을 이용한 사람과 자동차 이동을 구체적으로 구현 및 차별화)
날짜		10 월 16 일
12 회차	조덕진	LeapMotion 을 이용한 Object 배치 및 이동 수정작업 마무리 1 차 발표를 위한 Project 구성, ppt 제작, 내용정리
	김도현	미참석
	김영서	LeapMotion 을 이용한 Object 배치 및 이동 수정작업 마무리

		1 차 발표를 위한 Project 구성, ppt 제작, 내용정리
	윤희진	Unity 예제 학습 및 스크립트 공부 1 차 발표를 위한 Project 구성, ppt 제작, 내용정리
	이태경	Unity Build 작업을 수행시 발생하는 UI 의 좌표값 오류 수정작업 1 차 발표를 위한 Project 구성, ppt 제작, 내용정리
날짜		10 월 17 일
13 회차	조덕진	현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작
	김도현	현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작
	김영서	현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작
	윤희진	현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작
	이태경	현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작
날짜		10 월 18 일
14 회차	조덕진	구현되고 3D 오브젝트화 되어있는 UI 창들과 포함되어진 기능들을 LeapMotion controller 가 아닌 키보드와 마우스 작업으로 수행되도록 전환작업
	김도현	Data Object List 중 'Road' 파트 크기 조정 작업 해당 Object 배치시 자동적으로 정렬되는 기능 스크립트 구상/ 구현시작
	김영서	Object 배치와 이동을 LeapMotion 이 아닌 카메라 시점과 마우스를 이용하여 기능을 구현시키는 전환작업
	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	AI 의 행동에 대하여 Character 의 이동 루틴 구현(Navigation 을 이용한 사람, 자동차 이동)
날짜		10 월 19 일
15 회차	조덕진	LeapMotion 에서 키보드/마우스로 변환 작업 마무리 Object List 내의 Object 들의 크기조정 및 Renderer 조정/Collider 수정
	김도현	Object 중 Road 파트에 해당하는 Object 들의 크기 조정, 이를 배치할 때 자동 정렬되는 스냅핑 기능 구현을 위한 알고리즘 구상
	김영서	LeapMotion 에서 키보드/마우스로 변환 작업 마무리 Object List 내의 Object 들의 크기조정 및 Renderer 조정/Collider 수정
	윤희진	미참석
	이태경	AI 의 행동에 대하여 Character 의 이동 루틴 구현(Navigation 을 이용한 사람,

		자동차 이동)
날짜		10 월 22 일
16 회차	조덕진	전달 받은 Prefab 과 코드들을 하나의 프로젝트로 병합 및 Object 세부사항 수정 및 오류사항 확인 앞으로 구현해야할 동적 환경요소 변화에 대한 AI 대응 움직임들 등을 구상 및 작업분류
	김도현	미참석
	김영서	시뮬레이션에 필요한 동적 AI 객체에 해당하는 Object 들을 제작,수집 및 필요 스크립트 사항 추가/수정작업 앞으로 구현해야할 동적 환경요소 변화에 대한 AI 대응 움직임들 등을 구상 및 작업분류
	윤혜진	Unity 예제 학습 및 스크립트 공부
	이태경	AI 의 움직임을 위한 특정 카테고리 Object 들에 스크립트 및 요소 추가 앞으로 구현해야할 동적 환경요소 변화에 대한 AI 대응 움직임들 등을 구상 및 작업분류
날짜		10 월 23 일
17 회차	조덕진	상황변화 중 건물붕괴를 화재로 변경작업 마우스클릭을 이용한 Object 선택 및 해당 Object 에서의 상황변화 구현
	김도현	상황변화 중 태풍에 해당하는 작업 중 Object 크기 및 속성 변화 날씨변화 중 날씨 Object 의 생성 위치 및 크기 조정
	김영서	UI 인벤토리창을 이용하여 Object 생성시 Renderer 에 의해 발생한 오류수정
	윤혜진	Unity 예제 학습 및 스크립트 공부
	이태경	AI 에 해당하는 캐릭터들의 움직임/애니메이션을 수정 Rigidbody/Collider 추가작업과 Object 별로 속도조절
날짜		10 월 24 일
18 회차	조덕진	상황변화 건물 화재 선택 시 추가적으로 필요한 Object 의 생성 및 배치 기능을 위한 스크립트 구상 및 구현
	김도현	상황변화 태풍 선택 시 동적 요소 AI Object(사람)의 움직임을 위한 스크립트 구상 및 구현
	김영서	상황변화 날씨 선택 시 동적 요소 AI Object(사람)의 움직임과 Object 변화를 위한 스크립트 구상 및 구현

	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	동적요소 중 AI의 행동에 대하여 Character의 이동 루틴 구현(Navigation을 이용한 자동차 이동, 사람 AI와의 차별화)
날짜		10월 25일
19 회차	조덕진	상황변화 건물 화재 선택 시 동적 요소 AI Object(사람)의 움직임을 위한 스크립트 구상 및 구현
	김도현	상황변화 태풍 선택 시 동적 요소 AI Object(사람)의 움직임을 위한 스크립트 구상 및 구현
	김영서	상황변화 지진 선택 시 동적 요소 AI Object(사람)의 움직임과 Object 변화를 위한 스크립트 구상 및 구현
	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	동적요소 중 AI의 행동에 대하여 Character의 이동 루틴 구현(Navigation을 이용한 자동차 이동, 사람 AI와의 차별화)
날짜		10월 26일
20 회차	조덕진	상황변화(화재)에 따른 AI Object(사람)의 움직임 구현완료 및 추가적인 AI Object 생성 구현완료 다른 상황변화(태풍, 지진)에 대해 구현된 기능을 하나의 Project의 파일로 병합/ 이에 발생하는 오류 수정
	김도현	상황 변화(태풍)에 따른 AI Object(사람)의 움직임 및 효과 구현 완료 3 번째 예제 Background Terrain 제작 및 구상
	김영서	상황변화 지진 선택 시 동적 요소 AI Object(사람)의 움직임과 Object 변화를 위한 스크립트 구현 완료
	윤희진	미참석
	이태경	AI 움직임에 해당하는 애니메이션을 네비게이션에 적용하여 Road Test 시행 및 검출된 오류 수정
날짜		10월 29일
21 회차	조덕진	병합된 동적요소들을 전부 테스트하며 이에 발생하는 오류검출과 예외처리 및 수정 작업(동적 상황변화에 따른 위치 오류, 화재상황 오류)
	김도현	미참석
	김영서	병합된 동적요소들을 전부 테스트하며 이에 발생하는 오류검출과 예외처리 및 수정 작업(클릭이벤트, 캐릭터 충돌)

	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	병합된 동적요소들을 전부 테스트하며 이에 발생하는 오류검출과 예외처리 및 수정 작업(AI Navi 기능 오류)
날짜		10 월 30 일
22 회차	조덕진	AI Navi 병합과정 중 생긴 특정 Object 생성시 발생하는 오류 수정 유니티 작동 중에 현재 Scene 저장 기능 구현 및 스크립트 작성
	김도현	동적 상황요소 중 시간에 따른 조도 변화에 자동/수동 기능 구현
	김영서	유니티 작동 중에 현재 Scene 저장 기능 구현 및 스크립트 작성
	윤희진	Unity 예제 학습 및 스크립트 공부
	이태경	병합된 동적요소들을 전부 테스트하며 이에 발생하는 오류검출과 예외처리 및 수정 작업(AI Navi 기능 오류) 퍼지로직을 접목시키기 위한 알고리즘 및 구조 구상
날짜		10 월 31 일
23 회차	조덕진	구동 중 Scene 저장/불러오기 기능 구현 완료 동적 상황요소 중 수정되고 추가된 기능을 하나의 Project 로 병합 및 오류검출
	김도현	동적 상황요소 중 시간에 따른 조도 변화 자동/수동 기능 구현완료 3 차 Terrain 제작 진행
	김영서	구현한 Scene 저장/불러오기 기능 구현 완료, 추가적으로 발생하는 오류검출 및 수정 저장/불러오기 기능 구현 중 추가적인 기능 추가를 위한 디자인 및 제작시작
	이태경	주기적으로 작고 큰 오차가 발생하는 동적요소 AI 의 움직임(Navigation)기능의 오류 수정 및 최적화(Unity Tool 내부 문제로 추정) 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용)를 위한 알고리즘 및 구조 구상
날짜		11 월 01 일
24 회차	조덕진	전체적 최적화 작업들을 하나의 Project 로 병합(2 차 발표 Project 준비 통합본) 3 차 우선순위 작업 중 홀로그램 출력을 위한 작업 및 구상
	김도현	미참석
	김영서	기본적인 Scene 저장/불러오기 기능 구현완료 저장/불러오기 기능 추가 및 구현완료

	이태경	저장/불러오기 기능 구동을 위한 UI 제작 상황 변화에 따른 AI의 자율적 요소 추가(퍼지로직을 응용)를 위한 알고리즘 및 구조 구상
날짜		11 월 02 일
25 회차	조덕진	2 차발표를 위한 Project 최적화 및 발표준비(ppt 제작)
	김도현	미참석
	김영서	2 차발표를 위한 Project 최적화 및 발표준비(ppt 제작)
	이태경	2 차발표를 위한 Project 최적화 및 발표준비(ppt 제작)
날짜		11 월 05 일
26 회차	조덕진	2 차 발표
	김도현	2 차 발표
	김영서	2 차 발표
	이태경	2 차 발표
날짜		11 월 06 일
27 회차	조덕진	시뮬레이션 작동하는 On 모드에서 LeapMotion 을 추가하여 카메라의 이동 기능을 수행하도록 기존 Project 에 접목 병합 자체제작 카테고리 사용자에게 장애물 설치할 수있는 Object 를 추가/구현확인
	김도현	카메라의 이동 및 회전, 립모션 클릭 이벤트 시스템 기능에 대한 스크립트 정리
	김영서	시뮬레이션 작동하는 On 모드에서 LeapMotion 을 추가하여 카메라의 이동 기능을 수행하도록 기존 Project 에 접목 병합 자체제작 카테고리 사용자에게 장애물 설치할 수있는 Object 를 추가/구현확인
	이태경	상황 변화에 따른 AI의 자율적 요소 추가(퍼지로직을 응용)를 위한 알고리즘 및 구조/ 사전준비
날짜		11 월 07 일
28 회차	조덕진	홀로그램 HW 제작 구조 구상 LeapMotion 모드 및 VR 모드 구현완료/기존 PC 모드 프로젝트와 병합 완료
	김도현	홀로그램 HW 제작 구조 구상
	김영서	홀로그램 HW 제작 구조 구상 LeapMotion 모드 및 VR 모드 구현완료/기존 PC 모드 프로젝트와 병합 완료
	이태경	상황 변화에 따른 AI의 자율적 요소 추가(퍼지로직을 응용)를 위한 알고리즘

		및 구조/ 사전준비
날짜		11 월 08 일
29 회차	조덕진	홀로그램 출력을 위한 하드웨어 제작 시작
	김도현	미참석
	김영서	홀로그램 출력을 위한 하드웨어 제작 시작
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 시작
날짜		11 월 09 일
30 회차	조덕진	홀로그램 출력을 위한 하드웨어 제작 시작 프로젝트 Build 시 LeapMotion 의 좌표 설정 오류 수정 및 최적화
	김도현	미참석
	김영서	홀로그램 출력을 위한 하드웨어 제작 시작 프로젝트 Build 시 LeapMotion 의 좌표 설정 오류 수정 및 최적화
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 시작
날짜		11 월 12 일
31 회차	조덕진	홀로그램 제작을 위한 구조 설계 및 구성 재료 정리 홀로그램을 위한 영상 출력 및 카메라 구도 조정
	김도현	미참석
	김영서	홀로그램 제작을 위한 구조 설계 및 구성 재료 정리 홀로그램을 위한 영상 출력 및 카메라 구도 조정
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현/구상
날짜		11 월 13 일
32 회차	조덕진	홀로그램 Test Sample 및 1 차적으로 제작 완료
	김도현	미참석
	김영서	홀로그램 Test Sample 및 1 차적으로 제작 완료
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현/ Script 작성
날짜		11 월 14 일
33 회차	조덕진	홀로그램 출력 시험 및 홀로그램 크기/영상 출력 조정 완료 추가 동적 시뮬레이션(화재시 건물대피) 추가 및 구현을 위한 구상/환경 제작
	김도현	미참석

	김영서	홀로그램 출력 시험 및 홀로그램 크기/영상 출력 조정 완료 추가적으로 동적 상황변화요소(유성낙하) 추가 및 구현 완료
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현/ Script 작성
날짜		11 월 15 일
34 회차	조덕진	면접으로 인한 불참
	김도현	미참석
	김영서	3 차 발표를 위한 발표준비 및 영상 녹화/ 현재 프로젝트 최적화 시뮬레이션 Build 시 발생하는 LeapMotion 좌표오류 수정 및 대체
	이태경	상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 완료
날짜		11 월 16 일
35 회차	조덕진	현재까지 구현된 Project program 오류검출 및 최적화 수정된 LeapMotion 이벤트 처리를 기존 프로젝트에 수정/병합 현재 시뮬레이션 프로그램 프로젝트에 구현된 퍼지로직기능 적용/병합
	김도현	미참석
	김영서	네비웍스 면접
	이태경	네비웍스 면접
날짜		11 월 19 일
36 회차	조덕진	3 차 발표를 위한 발표준비 및 ppt 제작/ 3 차발표
	김도현	미참석
	김영서	3 차 발표를 위한 발표준비 및 ppt 제작/ 3 차발표
	이태경	3 차 발표를 위한 발표준비 및 ppt 제작/ 3 차발표
날짜		11 월 20 일
37 회차	조덕진	추가 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현
	김도현	미참석
	김영서	LeapMotion 세팅을 PC 모드에서 VR 모드로 전환시도
	이태경	AI 움직임 퍼지로직 수정(차량 충돌 및 사람과 차량 충돌 수정) 작업
날짜		11 월 21 일
38 회차	조덕진	완료보고서를 위한 회의 및 작성 AI 움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 수정된 퍼지로직 AI 움직임을 기존 Project 에 병합

	김도현	미참석
	김영서	완료보고서를 위한 회의 및 작성
	이태경	완료보고서를 위한 회의 및 작성
날짜		11 월 22 일
39 회차	조덕진	완료보고서를 위한 회의 및 작성 AI 움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현
	김도현	미참석
	김영서	완료보고서를 위한 회의 및 작성 현재 프로젝트 Test 실행 후 오류 검출 및 최적화
	이태경	완료보고서를 위한 회의 및 작성 현재 프로젝트 Test 실행 후 오류 검출 및 최적화
날짜		11 월 23 일
40 회차	조덕진	완료보고서를 위한 회의 및 작성 현재 프로젝트 Test 실행 후 오류 검출 및 최적화
	김도현	미참석
	김영서	완료보고서를 위한 회의 및 작성 현재 프로젝트 Test 실행 후 오류 검출 및 최적화
	이태경	완료보고서를 위한 회의 및 작성 현재 프로젝트 Test 실행 후 오류 검출 및 최적화

C. 회의록

1 차 회의록			
날짜	2018.09.27	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 조덕진		
안건	1. 작업 진행 상황 확인		
내용	1. Unity Untiy 를 사용한 전체적인 구조 설계, 필요 Object Data 조사 및 수집 2. Device I/O Control LeapMotion 개발환경 구축, 필요 SDK 설치, 설치 환경 테스트 3. 웹 서버 몽구스 서버 구축 및 AssetBundles 원리 파악		
2 차 회의록			
날짜	2018.09.28	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 윤혜진, 조덕진		
안건	1. 작업 진행 상황 확인		
내용	1. Unity - Unity 를 통한 UI 디자인 및 상세 카테고리 기능 추가[Object List(데이터 베이스) 추가] - 필요 Object Data 제작 및 예제 Terrain 구상 2. Device I/O Control - LeapMotion 기능별 탐구 및 예제 학습		

	<div>- 특정 제스처 코드 설계 회의 및 구상</div> <div>3. 웹 서버</div> <div>구축한 몽구스 웹서버 파일 경로 동기화</div> <div>AssetBundles 의 Sample Object 업로드 및 다운로드 테스트 수행</div>		
3 차 회의록			
날짜	2018.10.01	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 윤혜진, 조덕진		
안건	1. 작업 진행 상황 확인		
내용	<div>1. Unity</div> <div>- Unity 를 통한 UI_Inventory 디자인 제작 및 Object List 연동</div> <div>- 필요 Object Data 제작 및 예제 Terrain 구상</div> <div>- Sample Terrain 제작 및 구동 테스트</div> <div>- Object Shader 예제 학습</div> <div>2. Device I/O Control</div> <div>- LeapMotion 기능별 탐구 및 예제 학습</div> <div>- 특정 제스처 코드 분업화</div> <div>- 분업화한 제스처 코드 제작 및 구현</div>		
4 차 회의록			
날짜	2018.10.02	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 윤혜진, 조덕진		
안건	1. 작업 진행 상황 확인		

내용	1. Unity - List 내 Object 를 활성화, 게임 환경 생성 및 조정 - Sample Terrain 제작 마무리 - Object Shader 예제 학습		
	2. Device I/O Control - LeapMotion 카메라 줌인, 줌 아웃 코드 구상 - LeapMotion 카메라 상하시점 조정 기능 추가 - 분업화한 제스처 코드 제작 및 구현		
5 차 회의록			
날짜	2018.10.04	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 조덕진		
안건	1. 작업 진행 상황 확인		
내용	1. Unity - 시뮬레이션 제작 후 동적요소를 실행하는 ON/OFF 기능 및 UI 추가 - 세부 Object 별 AI 요소 및 동적 움직임 기능 알고리즘 구상 및 디자인 - Sample Terrain 제작 마무리 - Object Shader 예제 학습		
	2. Device I/O Control - LeapMotion 제스처별 기능 코드 작성 완료 - LeapMotion 으로 구현된 UI 와 연동 및 1 차적 테스트 진행 - 테스트 수행 후 발생하는 기능별 오차범위, 오류사항 발견 / 디버그 및 개선 방안 회의		
6 차 회의록			
날짜	2018.10.05	시간	오전 11:40 ~ 오후 12:00

참석자	김도현, 김영서, 이태경, 조덕진		
안건	1. 작업 진행 상황 확인		
내용	1. Unity - Shader 의 Sample Object 제작 및 Effect 코드 구현 - 2 차 Sample Terrain 구상 - UI 와 LeapMostion 기능의 병합 중 발견한 UI 의 추가/개선 사항 기능 추가 2. Device I/O Control - LeapMotion 을 활용, Camera 의 상하각도 조절 기능 구상 - 오류사항 해결 방안 협의 / 조정 - 구현된 UI 와 LeapMostion 기능의 병합 기능별 코드 분할 작업 - 손 떨림을 인한 UI 창 의 떨림효과 최소화 작업		
7 차 회의록			
날짜	2018.10.08	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 김영서, 이태경, 조덕진, 윤혜진		
안건	1. 작업 진행 상황 확인 2. 2 순위 작업 시작 3. 1 차 발표 준비		
내용	UI Unity - 수집한 Object Date 들과 제작한 1 차 Sample Terrain 을 인벤토리 UI 삽입 - DataBase Object 리스트에 적용 LeapMotion - Update 되고 작업이 완료된 Object 데이터들과 인벤토리 UI 를 추가 병합 작업 - 이에 해당하는 구현된 기능 테스트 및 오류 수정		

	시뮬레이션 내 특정 AI 행동에 대한 AI movement 의 해당 코드/ 알고리즘 구상		
	- 조덕진: 건물붕괴/지진		
	- 김도현: 날씨 전환		
	- 김영서: Skybox 변환을 이용한 계절변화		
	- 이태경: Navigation 을 이용한 사람, 자동차 이동		
8 차 회의록			
날짜	2018.10.10	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 이태경, 조덕진		
안건	1. 작업 진행 상황 확인 2. 2 순위 작업 시작		
내용	UI Unity & Leap Motion(조덕진) - 시뮬레이션에 필요한 Object Data 추가 수집/ 업데이트 및 분류 - Leap Motion 과 통합된 Sample 프로젝트와 수집된 Object 들을 DataBase 에 추가 및 통합 작업 시뮬레이션 내 특정 AI 행동에 대한 AI movement 의 해당 코드/ 알고리즘 구상 - 김도현: 날씨 전환, 재난/태풍 - 김영서: Skybox 변환을 이용한 계절변화 - 이태경: Navigation 을 이용한 사람, 자동차 이동		
9 차 회의록			
날짜	2018.10.11	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 이태경, 조덕진, 김영서, 윤혜진		
안건	1. 작업 진행 상황 확인 2. 2 순위 작업 시작		

내용	UI Unity & Asset Buddles(조덕진) - Object 들을 UI 인벤토리 DataBase 에 이미지 및 코드 추가 작업 완료 및 기능 테스트 - 필요한 3 가지 배경 Terrain 구상 및 제작 시작		
	시물레이션 내 특정 AI 행동에 대한 AI movement 의 해당 코드/ 알고리즘 구상 - 김도현: 날씨 전환, 재난/태풍 테스트 및 기능 시험 / 코드구현 - 김영서: Skybox 변환을 이용한 계절변화 - 이태경: Navigation 을 이용한 사람, 자동차 이동 - 윤혜진: Unity 예제 학습 및 스크립트 공부		
10 차 회의록			
날짜	2018.10.12	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 이태경, 조덕진, 김영서		
안건	1. 작업 진행 상황 확인 2. 2 순위 작업 시작		
내용	UI Unity - Object 의 이동할 시에 Object 의 표면을 기준으로 Slide 되는 개선방안 결정 및 스크립트를 구현하기 위한 탐색 및 구상 시물레이션 내 특정 AI 행동에 대한 AI movement 의 해당 코드/ 알고리즘 구상 - 구현된 동적 상황요소들(날씨, 태풍, 지진, 붕괴 계절 등)을 하나로 통합 - 위의 내용을 UI 를 제작 후 적용시켜 기능테스트 - Navigation 을 이용한 사람, 자동차 이동 구현		
11 차 회의록			
날짜	2018.10.15	시간	오전 11:40 ~ 오후 12:00
참석자	김도현, 이태경, 조덕진, 김영서, 윤혜진		

안건	1. 작업 진행 상황 확인		
내용	UI Unity - Sample Project 테스트후 오류 사항 개선방안 탐색 및 구상 - Object 리스트에 등록시킨 Object Data 들을 수정작업 시뮬레이션 내 특정 AI 행동에 대한 AI movement 의 해당 코드/ 알고리즘 구상 - 구현된 동적 상황요소들(날씨, 태풍, 지진, 붕괴 계절 등)을 하나로 통합 - 위의 내용을 UI 를 제작 후 적용시켜 기능테스트 - Navigation 을 이용한 사람, 자동차 이동 구현		
12 차 회의록			
날짜	2018.10.16	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서		
안건	1. 1 차 발표 준비 작업		
내용	UI Unity - Unity Build 작업을 수행시 발생하는 UI 의 좌표값 오류 수정작업 LeapMotion - LeapMotion 을 이용한 Object 배치 및 이동 수정작업 마무리 1 차 발표 준비 - 1 차 발표를 위한 Project 구성, ppt 제작, 내용정리		
13 차 회의록			
날짜	2018.10.17	시간	오전 11:40 ~ 오후 12:00

참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		
안건	1. 1 차 발표		
내용	1 차 발표 현재까지 진행된 프로젝트 정리 및 1 차 발표 ppt 제작		
14 차 회의록			
날짜	2018.10.18	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	Leap Motion I/O Controller - Main Controller 를 LeapMotion 에서 키보드/마우스로 변환 작업 - Object 배치와 이동을 LeapMotion 이 아닌 카메라 시점과 마우스를 이용하여 기능을 전환 3D Object - 수집되어 Data Object List 에 포함된 Object 들을 크기 조정 - Object 를 배치할 시 자동 정렬 스크립트 구상 디자인 AI - AI movement 의 해당 코드/ 알고리즘 구현(Navigation 을 이용한 사람, 자동차 이동) - AI 의 행동에 대하여 Character 의 이동 루틴 구현(Navigation 을 이용한 사람, 자동차 이동)		
15 차 회의록			
날짜	2018.10.19	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		

안건	1. 작업 진행 사항 확인		
내용	Leap Motion I/O Controller- LeapMotion 에서 키보드/마우스로 변환 작업 마무리- Object List 내의 Object 들의 크기조정 및 Renderer 조정/Collider 수정 3D Object- Object 중 Road 파트에 해당하는 Object 들의 크기 조정, - 이를 배치할 때 자동 정렬되는 스냅핑 기능 구현을 위한 알고리즘 구상 AI- AI movement 의 해당 코드/ 알고리즘 구현(Navigation 을 이용한 사람, 자동차 이동)- AI 의 행동에 대하여 Character 의 이동 루틴 구현(Navigation 을 이용한 사람, 자동차 이동)		
16 차 회의록			
날짜	2018.10.22	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		
내용	3D Object - 전달 받은 Prefab 과 코드들을 하나의 프로젝트로 병합 및 Object 세부사항 수정 및 오류사항 확인 - 시뮬레이션에 필요한 동적 AI 객체에 해당하는 Object 들을 제작,수집 및 필요 스크립트 사항 추가/수정 작업 AI - 동적 환경요소 변화에 대한 AI 대응 움직임들 등을 구상 및 작업분류 - AI 의 행동에 대하여 Character 의 이동 루틴 구현(Navigation 을 이용한 사람, 자동차 이동)		
17 차 회의록			
날짜	2018.10.23	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		

안건	1. 작업 진행 사항 확인		
내용	3D Object - 마우스클릭을 이용한 Object 선택 및 해당 Object 에서의 상황변화 구현 - UI 인벤토리창을 이용하여 Object 생성시 Renderer 에 의해 발생한 오류수정		
	AI - AI 에 해당하는 캐릭터들의 움직임/애니메이션을 수정 - Rigidbody/Collider 추가작업과 Object 별로 속도조절 - 상황변화 알고리즘 구성 및 코드 제작		
18 차 회의록			
날짜	2018.10.24	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	3D Object - 마우스클릭을 이용한 Object 선택 및 해당 Object 에서의 상황변화 구현 - UI 인벤토리창을 이용하여 Object 생성시 Renderer 에 의해 발생한 오류수정		
	AI - AI 에 해당하는 캐릭터들의 움직임/애니메이션을 수정 - Rigidbody/Collider 추가작업과 Object 별로 속도조절 - 상황변화 알고리즘 구성 및 코드 제작 - Navigation 을 이용한 자동차 이동, 사람 AI 와의 차별화		
19 차 회의록			
날짜	2018.10.25	시간	오전 11:40 ~ 오후 12:00

참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	AI - 상황변화에 따른 (지진, 태풍, 화재) AI Object 의 움직임 구상 및 구현 - 동적 요소 AI 의 움직임과 애니메이션 효과 구상 및 구현		
20 차 회의록			
날짜	2018.10.26	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	AI - 동적 요소 AI 의 움직임과 애니메이션 효과 구상 및 구현 - 게임 인공지능 요소 중 하나인 퍼지로직 알고리즘 구상 - 상황 변화에 따른 AI 코드를 3D Object 와의 배치 및 병합 및 버그 테스트 수행 3D Object - Background Terrain 제작 및 구상		
21 차 회의록			
날짜	2018.10.29	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서		

안전	1. 작업 진행 사항 확인		
내용	테스트 - 현 진행상황까지 구현된 기능 및 동정요소의 병합 및 최적화 - 3 차 진행 준비 및 2 차 진행 사항 추가 구현 기능 및 변화 요소 협의 및 토론		
22 차 회의록			
날짜	2018.10.30	시간	오전 11:40 ~ 오후 12:00
참석자	윤혜진, 이태경, 조덕진, 김영서		
안전	1. 작업 진행 사항 확인		
내용	AI - 동적 상황요소 중 시간에 따른 조도 변화 조정 기능 추가 테스트 - AI_Navi 병합 과정에서 생기는 오류 제어 및 최적화 - 병합된 동적 요소들의 테스트 및 버그의 예외처리 및 수정 작업 3D Object - Scene 저장 및 불러오기 기능 구상 및 기술 탐색		
23 차 회의록			
날짜	2018.10.31	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서, 김도현		

안건	1. 작업 진행 사항 확인		
내용	AI - 테스트 과정 중에 발생한 동적요소 AI 의 움직임(Navigation)기능의 오류 수정		
	테스트 - AI_Navi 병합 과정에서 생기는 오류 제어 및 최적화 - 병합된 동적 요소들의 테스트 및 버그의 예외처리 및 수정 작업		
	3D Object - 3 차 Terrain 제작 진행		
	- 구현한 Scene 저장/불러오기 기능 구현 완료 및 이에 발생하는 오류검출 및 수정		
24 차 회의록			
날짜	2018.11.01	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	AI - 동적 상황요소에 따른 AI 의 자율적 요소 추가를 위한 알고리즘 및 구조 구상		
	테스트 - 전체적 작업들의 병합 및 2 차 발표 준비		
	3D Object - Save/Load UI 디자인		
	- Scene 저장 및 불러오기 기능 구상 및 기술 구현		
25 차 회의록			

날짜	2018.11.02	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		
내용	2 차발표를 위한 Project 최적화 및 발표준비(ppt 제작)		
26 차 회의록			
날짜	2018.11.05	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 2 차발표		
내용	2 차발표		
27 차 회의록			
날짜	2018.11.06	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	AI - Human_Animator 수정 - 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용)를 위한 알고리즘 및 구조/사전준비 LeapMotion		

	<div>- 추가시킬 기능에 필요한 Leap motion 기능 정리</div> <div>3D Object</div> <div>- PC 버전으로 개발된 Project 에 일부 기능에 Leap Motion 기능을 접목</div> <div>- 자체제작 카테고리의 Object 추가 및 설정</div>		
28 차 회의록			
날짜	2018.11.07	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서, 김도현		
안건	1. 작업 진행 사항 확인		
내용	AI		
	<div>- 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현</div> <div>- Human_AI 퍼지 구상</div> <div>홀로그램</div> <div>- 홀로그램 출력을 위한 하드웨어 제작 구상</div> <div>- TrinusVR 기술을 이용해 프로젝트 VR 구현 및 기존 프로젝트와 병합</div>		
29 차 회의록			
날짜	2018.11.08	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		

내용	AI - 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 - Human_AI 퍼지 구상		
	홀로그램 - 홀로그램 출력을 위한 하드웨어 제작 시작		
30 차 회의록			
날짜	2018.11.09	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		
내용	AI - 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 - Human_AI 퍼지 구현 완료 및 테스트		
	홀로그램 & LeapMotion - 홀로그램 출력을 위한 하드웨어 제작 - Build 시 LeapMotion 의 좌표 설정 오류 수정 및 최적화		
31 차 회의록			
날짜	2018.11.12	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		

내용	AI - 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 - Car_AI 퍼지 구상		
	홀로그램 - 홀로그램 제작을 위한 재료 수집 및 제작 - 홀로그램을 위한 영상 출력 및 카메라 구도 조정		
32 차 회의록			
날짜	2018.11.13	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		
내용	AI - 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현 - Car_AI 퍼지 구상		
	홀로그램 - 홀로그램 제작을 위한 재료 수집 및 제작 - 홀로그램 Test Sample 및 1 차적으로 제작 완료		
33 차 회의록			
날짜	2018.11.14	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. 작업 진행 사항 확인		

내용	AI		
	- 상황 변화에 따른 AI 의 자율적 요소 추가(퍼지로직을 응용) 구현		
	- Car_AI 퍼지 구현 완료 및 테스트		
내용	- 추가 동적 시뮬레이션(화재시 건물대피) 추가 및 구현을 위한 구상/환경 제작		
	홀로그램		
	- 홀로그램 출력 시험 및 홀로그램 수정작업		
34 차 회의록			
날짜	2018.11.14	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 김영서		
안건	1. 3 차 발표		
내용	3 차 발표를 위한 발표준비 및 영상 녹화/ 현재 프로젝트 최적화		
35 차 회의록			
날짜	2018.11.14	시간	오전 11:40 ~ 오후 12:00
참석자	조덕진		
안건	1. 3 차 발표		

내용	3 차 발표를 위한 발표준비 및 PPT 작성		
36 차 회의록			
날짜	2018.11.19	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서, 김도현		
안건	1. 3 차 발표		
내용	3 차 발표		
37 차 회의록			
날짜	2018.11.20	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. AI 2. LeapMotion		
내용	1. AI - 추가 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 - AI 움직임 퍼지로직 수정(차량 충돌 및 사람과 차량 충돌 수정) 작업 2. LeapMotion		

	- LeapMotion 세팅을 PC 모드에서 VR 모드로 전환시도		
38 차 회의록			
날짜	2018.11.21	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. AI 2. Holo_City 완료 보고서 작성		
내용	1. AI - AI 움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 - 수정된 퍼지로직 AI 움직임을 기존 Project 에 병합 2. Holo_City 완료 보고서 작성 - 완료보고서를 위한 회의 및 작성		
39 차 회의록			
날짜	2018.11.22	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. Holo_City 완료 보고서 작성		

내용	1. Holo_City 완료 보고서 작성 - 완료보고서를 위한 회의 및 작성 - 개인 개발일지 및 보고서 작성		
40 차 회의록			
날짜	2018.11.23	시간	오전 11:40 ~ 오후 12:00
참석자	이태경, 조덕진, 김영서		
안건	1. Holo_City 완료 보고서 작성		
내용	1. Holo_City 완료 보고서 작성 - 완료보고서를 위한 회의 및 작성		

D. 개인개발 일지

i. 김영서 개발일지

개발일지			
날짜	2018.09.27	작성시간	18:00
이름	김영서		
개발과제	Leap Motion 개발환경 구축 Leap Motion 예제 학습 Leap Motion 구조 파악		
내용	유니티 개발툴 Leap_Motion_Core_Assets_ 4.4.0 Leap_Motion_Interaction_Engine_1.2.0 Leap_Motion_Hands_Module_2.1.4 립모션설치후 유니티를 이용해 립모션이 작동하는지 확인해보고 립모션에서 제공해주는 예제 몇가지를 실행시켜 어떻게 작동하는지 알아보았다. 직접 립모션 프로그래밍했을 때 처음 기획하면서 생각했던것과는 조금 다르게 작동하는걸 확인했다. 오브젝트를 원하는 곳에 위치시킬 때, 오브젝트를 잡아서 이동은 되지만 오브젝트가 충돌시 날아가거나 원하는 위치에 가도록 세심한 이동이 불가능 했다.		

개발일지			
날짜	2018.09.28	작성시간	18:00
이름	김영서		
개발과제	립모션 구체적인 동작 정의		
내용	<ol style="list-style-type: none"> 오브젝트 선택 <ul style="list-style-type: none"> - Trigger 오브젝트 확대, 축소 <ul style="list-style-type: none"> - 오브젝트를 양손으로 잡아늘리고 줄인다. 오브젝트 이동 <ul style="list-style-type: none"> - 오른손 엄지-좌, 검지-앞, 우-새끼, 뒤- 검+중 오브젝트 삭제 <ul style="list-style-type: none"> - 선택후 삭제메뉴출력 카메라 줌인, 줌아웃, 회전 <ul style="list-style-type: none"> - 왼손-주먹, 오른손-검지와 엄지만펴고 나머지는 접는다. 시뮬레이션 저장 <ul style="list-style-type: none"> - 손으로 카메라모양 		

개발일지			
날짜	2018.10.01	작성시간	20:00
이름	김영서		
개발과제	립모션을 이용한 오브젝트이동, 배치		
내용	<p>처음에 립모션에서 제공하는 Interaction Engine 을 이용해 오브젝트를 잡고 배치시킬 수 있게 하였다. 하지만 오브젝트들의 충돌을 해결하지 못했고, 다른사람들과 합치는 과정에서 사용할 수 없어서 새로 만들게 되었다. 손의 좌표와 오브젝트좌표를 이용해 만들어보았고, 실행까지 되는것을 확인했다.</p>		

개발일지			
날짜	2018.10.02	작성시간	19:30
이름	김영서		
개발과제	1. Leap Motion Grap 모션 구현 2. 카메라 상하시점이동, 시점이동모션 변경		
내용	<p>기존의 다른 팀원이 구현한 시점이동의 경우 양손을 같이 움직여야 시점이동이 가능하고, 좌우이동만 가능하고 상하로는 시점 전환이 불가능했다. 먼저, 한손으로 움직일 수 있게하기 위해서 제스처를 새로 정의해 왼손은 주먹을 쥐고 있는 상태로 오른손의 상하좌우이동을 통해서 손의 방향대로 시점이 변하게 만들었다.</p> <p>동작을 만들고 나서 문제가 하나 더있었는데, 좌우로만 이동할 때는 카메라의 각도가 문제가 없지만네방향으로 이동하게 만들어보니 좌표값이 서로 섞이게 되서 각도가 틀어지게 되었다. 여러가지 방법을 시도해 보았으나, 아직 해결방법을 찾지 못했다.</p>		

개발일지			
날짜	2018.10.04	작성시간	19:30
이름	김영서		
개발과제	카메라 상하시점 변경		
내용	<p>카메라 Rotation 의 x 축이 바뀐상태로, y 축을 이동시키면 z 축이 같이 이동하게 되어 생각한것과 다른각도가 되는 문제가 있는데 해결방법을 찾지못했다. 처음에는 오일러각을 이용했는데 안되서 쿼터니언을 이용해 y 축이 변화할때에 z 값과 x 값을 고정시키고, x 축이 이동할때는 z 값과 y 값을 고정시켰지만 생각처럼 되지않아서 동작할때마다 45 도씩 위아래로 틀어지게 만들기로 했다.</p>		

개발일지			
날짜	2018.10.05	작성시간	18:00
이름	김영서		
개발과제	Leap Motion 초기작업 완료		
내용	<p>립모션 카메라회전,이동 작업완료후에 팀원들과 나눠서 코딩한 립모션동작을 하나로 합쳐서 제대로 되는지 확인하고, 문제가 있는 부분을 찾아 회의를 통해 어떤식으로 고쳐야할지 생각해보았다.</p>		

개발일지			
날짜	2018.10.08	작성시간	18:00
이름	김영서		
개발과제	계절, 시간에 따른 변화적용		
내용	<p>1. 계절 : 4 계절로 나누어 계절별 날씨와 계절에 따른 물체들의 변화를 표현한다.</p> <p>2. 시간 : 시간의 흐름에 따라 변하는 태양과 그림자의 위치변화, 늦은시간이 될수록 어두워지고 가로등과 건물의 불빛이 켜지게 해 아침,점심,저녁을 나타낼수 있도록 한다.</p> <p>3. 날씨 : 비,눈등의 날씨에 따른 하늘의 변화</p> <p>일단 위세가지를 목표로 어떻게 만들어야 할지 생각해보고, 자료수집을 하였다.</p>		

개발일지			
날짜	2018.10.11	작성시간	20:00
이름	김영서		
개발과제	날씨,시간변화 완료		
내용	<p>시간 : 슬라이더바를 이용해 시간이 변할때, 빛이 들어오는 각도를 조절해서 태양을 표현하고, 조도를 조정하게 하였다. 정오를 기준으로 가운데로 놓고 왼쪽은 아침, 오른쪽은 저녁으로 나타내보았다. 하늘의 경우 skybox 의 밝기 조절을 통해 시간이 바뀔 때 같이 밝고, 어두워지게 만들었다.</p> <p>날씨 : 비와 눈이 오면 어두워지고, 흐린날씨로 변화한다.</p> <p>계절 : 당장 필요한 기능이 아니라고 생각되어 일단 구현하지 않고 보류</p>		

개발일지			
날짜	2018.10.12	작성시간	19:00
이름	김영서		
개발과제	Object 배치 변경, 개선방안 구상		
내용	<p>프로그램 테스트과정에서 오브젝트 설치와 이동이 너무 불편하고, 원하는 위치에 가지 않아 팀원들과 이야기해본 결과, 새로만드는 방향으로 결정했다. 기존의 물체를 손으로 집어서 이동하는 방식에서 손가락으로 위치시킨 곳에 오브젝트가 이동하는 식으로 만들기로 하였다. 일단 왼손 손가락으로 위치를 가리키면 그곳에 흐릿한 건물의 이미지가 나오고, 그 위치에 입력을 주면 보이는 그대로 생성 되게 하고 바닥의 표면을 따라서만 생성되게 하기로 했다.</p>		

개발일지			
날짜	2018.10.15	작성시간	19:00
이름	김영서		
개발과제	Object 배치 변경, 개선		
내용	<p>1. 반투명한 이미지 생성 : 오브젝트의 모양과 설치될 이미지를 미리 보여주기위해 오브젝트를 투명하게 만들었다.</p> <p>2. 생성위치 : 바닥에 RayCast 를 쏘고 바닥에 부딪혔을 때, 그위치를 받아와서 반투명한 오브젝트를 생성해 어떻게 설치될지 미리 볼수 있게 하였다.</p> <p>3. 오브젝트 생성 : 미리보기한 이미지와 같은곳에 똑같은 오브젝트가 생성된다.</p> <p>오브젝트 생성시에 각도가 원하는대로 되지않아 오브젝트들의 각도를 조절해야하는 문제가 생겼다.</p> <p>또한, 물체가 생성될때 바닥에서 튕겨나가는 문제도 해결해야한다.</p>		

개발일지			
날짜	2018.10.16	작성시간	19:00
이름	김영서		
개발과제	Object 배치 변경, 개선 PPT 발표 자료 준비		
내용	<p>어제 했던 오브젝트배치를 마무리하고, 1 차발표를 위해 제대로 작동하는지 테스트를 진행했다.</p> <p>1 차발표에 들어갈 내용들을 생각하고 자료를 준비해서 PPT 를 만들었다.</p>		

개발일지			
날짜	2018.10.17	작성시간	18:00
이름	김영서		
개발과제	프로젝트 1 차 발표		
내용	1 차적인 목표로 한 내용들에 대해서 PPT 를 작성해서 발표했다. 미흡한점들은 2 차발표때 까지 보완이 필요하다.		

개발일지			
날짜	2018.10.18	작성시간	21:00
이름	김영서		
개발과제	<p>립모션 -> 마우스,키보드 변경작업</p> <ol style="list-style-type: none"> 1. 카메라 시점이동 2. 오브젝트 선택,이동 		
내용	<p>1 차발표 이후에 립모션으로 계속 프로젝트를 진행하기가 힘들어서 일단은 마우스와 키보드로 바꾸는 방향으로 하고 립모션은 부가적인 기능으로 사용하기로 결정했다. 그래서 일단 기존의 립모션으로 동작했던것들을 바꾸는 작업을 했다.</p> <p>카메라 이동 : 기존의 립모션을 이용한 이동에서 키보드(W,S,A,D)방향으로 이동하도록 변경했다. 마우스의 움직임에 따라 카메라의 각도가 변경된다.</p> <p>오브젝트 이동 : 설치된 오브젝트를 마우스 클릭하면 클릭한 오브젝트가 선택되고 정보가 나타나고, 마우스의 이동에 따라 위치가 변하고 다시한번 클릭하면 이동한 위치에 재배치된다.</p>		

개발일지			
날짜	2018.10.19	작성시간	18:00
이름	김영서		
개발과제	LeapMotion 에서 키보드/마우스로 변환 작업 마무리 Object List 내의 Object 들의 수정작업		
내용	NUI 에서 GUI 로 모두 변환한뒤에 제대로 작동하는지 확인했다. 동작에는 문제가 없으나 오브젝트가 설치가 이상하게되거나 설치가 안되는것들을 확인하고, 크기가 맞지않는 오브젝트들을 전부 비율이 맞도록 수정했다.		

개발일지			
날짜	2018.10.22	작성시간	20:30
이름	김영서		
개발과제	현재까지 진행된 UI, Cotroll 요소 와 기본적인 AI 움직임 병합과정		
내용	<p>오브젝트 크기를 맞춰서 팀원에게 넘겨줬는데 크기가 이상하게 변해서 다시한번 수정한뒤에 넘겨줬다.</p> <p>시간변화에 따른 자동차들의 불빛변화와 자동차오브젝트 수집, 가로등 불빛수정</p>		

개발일지			
날짜	2018.10.23	작성시간	21:00
이름	김영서		
개발과제	오브젝트 생성위치 수정 화재시 소방차 출동		
내용	<p>땅속에 파고들어 생성되거나 각도가 이상하게 나오는 오브젝트들을 찾아서 땅위에 제대로 생성되도록 수정했다. 오브젝트들이 수정을 했음에도 원하는대로 생성이 되지않아서 안되는것들을 찾아서 생성할 때 수정되어 생성될 수 있도록 코드를 수정했다.</p> <p>건물에 화재가 발생 했을 때 주변에 사람이 다가오지 못하고 소방차와 앰불런스가 출동하게 하려했으나 의욕이 생기지않아서 제대로 하지못했다. 내일은 새로운 마음으로 다시 시도해보도록 해야겠다.</p>		

개발일지			
날짜	2018.10.24	작성시간	18:00
이름	김영서		
개발과제	상황 변화(날씨)에 따른 AI Object(사람)의 변화와 움직임 구상/구현		
내용	<p>날씨 변화(비, 눈 등)에 따른 캐릭터들의 상태변화를 나타냈다. 비가 올때 캐릭터가 우산을 쓰게하려고, 처음에는 비가 오면 캐릭터가 우산을 들고있는 모션을 제작해보려 했으나. 반나절동안 시도해보고 안하는게 좋겠다고 결정했다. 그래서 캐릭터에 우산모양의 오브젝트를 만들어 둔 뒤에 비가 오면 우산이 캐릭터 위에 우산을 쓰고있는것처럼 생성되게 하였다.</p>		

개발일지			
날짜	2018.10.25	작성시간	18:00
이름	김영서		
개발과제	상황 변화(지진)에 따른 AI Object(사람)의 변화와 움직임 구상/구현		
내용	<p>지진이 발생했을 때, 사람들이 어떻게 대처해야 할지 생각해 보았다. 일단 지진이 발생하면 주변에 건물이 없는 공터로 가는게 좋다고 생각했다. 사람들이 움직이다가 지진이 발생하면 공원,도로 같은 건물에서 떨어져 있는 곳으로 가게하기 위해서 캐릭터가 어떻게 움직이고 목표한곳으로 가게되는지 공부했다. 가장 가까운 위치에 있는 공터들을 캐릭터가 찾아서 그위치로 캐릭터가 이동하는 것으로 결정했다.</p>		

개발일지			
날짜	2018.10.26	작성시간	18:00
이름	김영서		
개발과제	상황 변화(지진)에 따른 AI Object(사람)의 변화와 움직임 구상/구현		
내용	<p>어제에 이어서 지진이 발생했을 때, 사람들의 움직임을 구현했다. 우선 공원, 도로를 찾아서 그중에서도 가장 가까운 위치에 있는 곳으로 사람들이 달려서 대피하도록 만들었다. 그리고 지진이 끝나게 되면 사람들이 다시 평소처럼 걸어다니도록 만들었다.</p>		

개발일지			
날짜	2018.10.29	작성시간	18:00
이름	김영서		
개발과제	<p>현재까지 구현된 기능과 동적요소를 전부 병합 및 최적화 추가 구현할 기능 또는 변화요소를 협의 및 토론</p>		
내용	<p>회의를 통해 2 차발표가 얼마 남지않아서 더 이상 기능추가는 하지않고, 이미 구현중인것들은 이번주내로 마무리하고, 구현한 것들을 합친후에 제대로 동작하는지 확인하기로 했다. 말아서 하는것중에 지진은 마무리 했고, 저장-불러오기, UI 창 뒤로 클릭이 되는 문제를 해결하기로 했다.</p> <p>UI 클릭문제는 쉽게 문제를 해결했고, 저장을 어떻게 해야할지 찾아봤는데 복잡한 기능이 필요하지 않아서 찾아본 방법중에 json 을 이용해 오브젝트의 정보들을 저장하고 정보를 불러와 생성시키는 쪽으로 결정했다.</p>		

개발일지			
날짜	2018.10.30	작성시간	18:00
이름	김영서		
개발과제	유니티 작동 중에 현재 Scene 저장/불러오기 기능구상 및 기술 탐색		
내용	<p>저장을 하기위해서 일단 화면상의 모든 오브젝트의 정보를 찾아 json 형식으로 변환해 파일로 저장했다. 모든 오브젝트를 저장하려고 하니 저장이 안되는것들이 있어서, 안되는 이유를 찾아서 예외처리를 통해 저장되게 했다. 오브젝트 정보를 하나의 리스트에 담아서 json 파일로 저장한 뒤에 불러오기를 할때는 저장된 정보를 가지고, 그에 맞는 오브젝트에 정보를 입력해 생성하게 했다. 저장-불러오기는 되었고, 내일은 저장에 관련된 UI 를 구현해야 한다.</p>		

개발일지			
날짜	2018.10.31	작성시간	18:00
이름	김영서		
개발과제	구현한 Scene 저장/불러오기 기능 구현 완료, 추가적으로 발생하는 오류검출 및 수정 저장/불러오기 기능 구현 중 추가적인 기능 추가를 위한 디자인 및 제작시작		
내용	<p>저장-불러오기 기능에 필요한 UI 들을 만들었다. 저장버튼, 불러오기버튼, 저장된 파일목록을 보여줄 수있는 팝업창을 만들었다. 저장버튼을 누르면 저장할 파일의 이름을 입력받는 팝업창이 뜨고, 입력후 확인을 누르면 저장이 완료되게 했다. 저장할 때 같은이름으로 덮어쓰기를 하면 새로만들어 지지 않고 뒤에 추가가 되는 문제가 있다. 불러오기버튼은 불러오기는 잘되지만 파일목록을 선택해서 불러올수가 없어서 저장된 파일목록을 불러오게 했고, 불러온 파일을 표시하는데 크기조절이 필요하다.</p>		

개발일지			
날짜	2018.11.01	작성시간	20:00
이름	김영서		
개발과제	<p>기본적인 Scene 저장/불러오기 기능 구현 저장/불러오기 기능 추가 및 구현</p>		
내용	<p>어제 구현한 UI 에서 모자란부분들을 수정했다. 저장할때 새로저장이 안되는 문제를 해결했고, 파일목록을 불러와 파일명,생성시간이 표시되게 했다. 파일목록에서 선택한 파일의 색깔을 바꿔게 해서 어떤 파일을 선택했는지 알 수 있게했다. 파일목록을 불러올때 목록이 새로고침이 되지않고 계속 추가되었는데 그 문제까지 전부 해결하고, UI 디자인을 이전까지 했던것들과 통일 시킨후에 전부 한곳에 모아 합친후에 결과까지 확인했다.</p>		

개발일지			
날짜	2018.11.02	작성시간	18:00
이름	김영서		
개발과제	<p>프로그램 빌드 및 오류수정 2 차발표준비</p>		
내용	<p>프로그램을 빌드 후에 제대로 작동하는지 확인하고, 팀원들과 발표 파워포인트를 작성했다.</p>		

개발일지			
날짜	2018.11.05	작성시간	18:00
이름	김영서		
개발과제	2 차 발표		
내용	2 차 발표를 한뒤에 문제점들과 이제 고쳐야할점과 추가할것들에 대해 논의했다.		

개발일지			
날짜	2018.11.06	작성시간	18:00
이름	김영서		
개발과제	PC 버전으로 개발된 Project 에 일부 기능에 Leap Motion 기능을 접목 자체제작 카테고리의 Object 추가 및 설정		
내용	<p>지금 까지 있는 기능에 립모션을 추가해 립모드를 따로 하나 더 구현했다. 예전에 해놓았던 립모션기능들이 제대로 동작해서 마우스모드와 립모션모드로 따로 나눠 진행하는 식으로 했다. 립모션으로 모든 기능을 사용하는데에는 무리가 있어서 일부기능만 립모션모드에서 사용가능 하도록 했다.</p>		

개발일지			
날짜	2018.11.07	작성시간	18:00
이름	김영서		
개발과제	<p>홀로그램 출력을 위한 하드웨어 제작 구상</p> <p>TrinusVR 기술을 이용해 프로젝트 VR 구현 및 기존 프로젝트와 병합</p>		
내용	<p>지금까지 오브젝트 설치할때 설치한뒤에 선택을해서 각도를 돌려야 해서 불편하고 계속 마음에 들지 않았는데 2 차발표 이후에 수정작업을 하는김에 기능을 넣었다. 이제 오브젝트 설치시에 R 키를 누르면 각도가 45 도씩 바뀌게 만들어 설치가 더 쉬워졌다.</p> <p>TrinusVR 이 제대로 적용되는지 확인해봤다. 저번에 시도 했을때는 제대로 작동이 안됐는데 지금은 아주 잘 동작하는것을 확인했다.</p> <p>홀로그램제작을 빨리해야하는데 언제 제작할것인지 정하고 샘플을 만들어보기로 했다.</p>		

개발일지			
날짜	2018.11.08	작성시간	18:00
이름	김영서		
개발과제	홀로그램 출력을 위한 하드웨어 제작		
내용	<p>홀로그램 제작을 위해서 모니터크기에 맞는 아크릴판을 구매했다. 처음에는 모니터 4 개를 이용해 커다란 홀로그램장치를 만드려고 했으나 맞는크기의 아크릴판을 구하기도 힘들고 틀제작에 어려움이 있어서 그냥 하나의 모니터에 출력하는 식으로 하기로 했다.</p> <p>립모션에 문제가 있어서 문제를 해결하고 홀로그램제작을 하기로 했다. 빌드시에 UI 의 좌표가 어긋나는 문제가 있다. 빌드하기전에는 문제가 없는데 빌드시에만 문제가 생긴다.</p>		

개발일지			
날짜	2018.11.09	작성시간	18:00
이름	김영서		
개발과제	프로그램 빌드시 LeapMotion 문제 해결		
내용	<p>립모션기능은 거의 다 완성을 했지만, 빌드하기 전에는 문제가 없이 잘되는데 빌드를 하게 되면 자꾸 UI 의 위치와 손의 위치가 맞지않는 문제가 있다. 어제밤부터 계속해서 해결하려고 많은 방법을 시도해 보았지만 해결을 하지못했다. 일단 시간낭비라고 판단해서 빌드하지 않고 시연하는 방향으로 정했지만 시간이 남으면 다시 시도해봐야한다.</p>		

개발일지			
날짜	2018.11.12	작성시간	20:00
이름	김영서		
개발과제	홀로그램 장치 제작		
내용	<p>오전에 포트폴리오를 작성하고 오후에는 홀로그램 출력할 장치를 만들었다. 처음시도에 실패해서 하루종일 붙잡고 새로 만들어서 완성시켰다. 생각보다 만족스럽게 홀로그램이 나오지 않아서 좀 더 잘나올수 있는 방법을 생각해봐야 겠다. 내일부터는 다시 유니티프로그래밍으로 돌아가서 화재발생시 건물내부 대피 시뮬레이션을 구현해야한다.</p>		

개발일지			
날짜	2018.11.13	작성시간	20:00
이름	김영서		
개발과제	상황 시뮬레이션 추가(유성낙하)		
내용	<p>자연재해 메뉴탭에 유성낙하를 추가 했다. 마우스로 클릭한 위치에 유성우가 생성되어 위에서 부터 빠른속도로 떨어져 내린다. 도로와 맵을 제외한 모든것들은 운석에 맞게되면 모두 날아가고 불타오르게 된다. 립모션모드의 경우에는 지정된 위치에 유성이 떨어져 내리도록 만들었다. 지정된 위치이긴 하지만 유성이 많이 생성되면서 랜덤한 위치에 생성이 되서 한곳에 몰리지는 않는다.</p>		

개발일지			
날짜	2018.11.14	작성시간	19:40
이름	김영서		
개발과제	빌드시 립모션 오작동 오류 해결		
내용	<p>이전부터 계속 사용해 오던 립모션의 기능으로는 도저히 오류를 해결하지 못할것 같아서 아예 처음으로 돌아가서 전부 다시 만들었다. 마우스 모드에서 할 수 있는 기능들을 모두 사용하기에는 무리가 있고, VR 로 작동시킬 때 직접 상황을 움직이면서 볼 수 있도록 시뮬레이션 기능정도만 추가했다. 모든 기능을 넣고 싶긴하지만 다른 기능을 넣으려면 너무복잡해져서 시간문제도 있고, 립모션이 생각보다 컨트롤하는게 쉽지가 않아서 일단은 마무리하기로 했다.</p>		

개발일지			
날짜	2018.11.15	작성시간	18:00
이름	김영서		
개발과제	3 차발표 PPT 제작 UI 수정 홀로그램 동시출력확인		
내용	<p>이전에 사용한 PPT 에 변경한것들에 대해 수정이 필요해서 사진을 다시 찍고 3 차발표에 사용할 PPT 제작</p> <p>UI 크기가 맞지 않는것들이 있어서 UI 를 전체적으로 크기와 디자인을 수정했다.</p> <p>빌드시에 한쪽화면은 프로그램화면이 나오게 하고 한쪽화면은 홀로그램을 출력할수 있는 화면으로 나오게 했다.</p>		

개발일지			
날짜	2018.11.16	작성시간	19:00
이름	김영서		
개발과제	네비웍스 견학		
내용	네비웍스 견학		

개발일지			
날짜	2018.11.19	작성시간	19:00
이름	김영서		
개발과제	3 차 발표		
내용	3 차 발표		

개발일지			
날짜	2018.11.20	작성시간	18:00
이름	김영서		
개발과제	립모션 모드 PC -> VR 로 변경		
내용	<p>현재 립모션이 손인식이 PC 형태로 되어있어서 립모션을 책상에 놓고 그 위에 손을 올려서 작동을 하게 되었는데 립모드를 키고 VR 로 시연할때에 바닥에 놓고 하면 확실히 보여주기가 애매해서 립모션이 제공해주는 VR 모드를 이용해 VR 기기에 립모션을 부착해서 사용하려고 했지만, VR 형태로 바꾸게 되면 손인식을 제대로 못하고 인식이 되더라도 위치가 자꾸 이상한 곳으로 날아가거나 버튼이 클릭이 안되서 계속 하는건 시간낭비라고 생각되어 원래 하던대로 하기로 했다.</p>		

개발일지			
날짜	2018.11.22	작성시간	18:00
이름	김영서		
개발과제	문서작성 완료		
내용	<p>어제부터 이어서 하던 문서작성을 마무리 하고 다같이 확인했다. 이제 남은 시간에는 프로젝트를 계속 진행하고 나중에 수정이 필요하면 수정을 하면 될 것 같다. 이제 기능을 추가하기보다는 실행해보고 여러가지를 시험해보고 이상하거나 미흡한부분들에 대해 수정을 해야겠다.</p>		

개발일지			
날짜	2018.11.23	작성시간	18:00
이름	김영서		
개발과제	프로그램 최적화		
내용	<p>프로그램을 오랫동안 실행해서 확인해본적은 없는데 계속 하다보니까 미흡한부분들이 발견되었다.</p> <p>자잘한 문제들이 많아서 몇가지 문제를 수정했다.</p>		

ii. 조덕진 개발일지

개발일지			
날짜	9/27/2018	작성시간	18:15:00 PM
이름	조덕진		
개발과제	1. Asset Bundle 개념 이해 및 원리 파악 2. Asset Bundle에 이용할 웹서버 조사, 결정 및 설치		
내용	1. Asset Bundle 개념 이해 및 원리 파악 전체 프로젝트 목표 및 방향을 고려하고 회의토론을 하고 난 후에 DataBase를 구축할 필요가 없다고 판단이 되었다. 주 개발환경이 Unity이기 때문에 간단한 DataBase를 대체할 수 있도록 Asset Bundle 을 사용하기로 하였다. 에셋 번들은 Unity에서 원하는 에셋을 포함하여 익스포트할 수 있는 파일을 의미한다. 이러한 파일은 전용 압축 포맷을 사용하여 플레이어에서 필요로 할 때 로드 할 수 있습니다. 이는 콘텐츠, 텍스처, 오디오 클립 또는 씬 전체를 스트리밍할 수 있다. 에셋 번들은 응용 프로그램에 콘텐츠를 다운로드하는 것을 단순화하도록 설계되어 있다. 유니티에서 제공하는 코드를 이용해 에셋번들 생성과 빌드를 하였다.		
	2. Asset Bundle에 이용할 웹서버 조사, 결정 및 설치 캐싱을 이용한 에셋번들 다운로드 및 로딩을 위해 간단한 설치로 웹서버 이용할 수 있는 몽구스 웹서버를 선택 및 설치 하였다.		

개발일지			
날짜	9/28/2018	작성시간	18:02:00 PM
이름	조덕진		
개발과제	1. 설치한 로컬 웹 서버를 이용한 구축 및 동기화 2. Leap Motion Gesture기능 코드작성		
내용	<p>1. 설치한 로컬 웹 서버를 이용한 구축 및 동기화 몽구스 웹서버와 에셋번들 데이터를 업로드/다운로드 할 파일의 경로를 동기화 설정하였다. 파일의 경로를 public Object로 입력받아 직접 입력하여 설정하였다. 이 과정에서 파일명이 다르게 되면 수행되지 않기 때문에 수차례 시행착오가 있었지만 동기화 시킬 수 있었다. 이를 테스트하기 위하여 Sample Object(3D Object, Cube)를 이용하였고 유니티에서 제공하는 캐싱 방법 코드를 이용하여 업로드 및 다운로드 테스트를 확인하였다.</p> <p>2. 로컬 웹서버를 이용한 AssetBundle 설계 및 테스트 과정이 일정보다 일찍 끝났다. 하지만 Leap Motion 센서를 이용한 기능 구현과 컨트롤러로서의 대체하는 과정이 계획보다 길어지고 인원보충이 필요하다고 생각을 하였다. 그래서 역할을 부담하여 Leap Motion을 이용한 특정 제스처를 구상하고 기능 코드를 작성하였다. 1차적으로 Sample Object를 이용해서 크기를 조정하고 카메라의 시점이동, 위치 이동하는 코드를 구상하였고 기능 코드 제작에 들어갔다.</p>		

개발일지			
날짜	10/1/2018	작성시간	19:10:00 PM
이름	조덕진		
개발과제	<p>1. Leap Motion 센서를 활용하여 특정 모션을 이용한 Object를 카메라 시점이동, 위치이동 기능 코드 구현</p> <p>2. Object의 크기를 모션으로 제어 및 조정하는 기능 구상 및 코드 제작</p>		
내용	<p>1. Leap Motion 센서를 활용하여 특정 모션을 이용한 Object를 카메라 시점이동, 위치이동 기능 코드 구현</p> <p>- 회의를 통해 임시로 결정한 기능별 손동작을 바탕으로 해당 모션을 취했을 때의 기능을 구현할 수 있는 코드 작성을 시작했다.</p> <p>원손을 오므린 상태에서 카메라의 좌우 시점이동을 구현을 하였다. 초기에 상하 역시 이동이 가능하게 제작하였는데, 카메라가 180도 회전 되어 버리고 기울어지는 상황을 막기 위해 z의 rotation를 0으로 고정시켜 가며 테스트를 하였다. 하지만 카메라의 x, z의 Rotation이 합쳐져서 계산이 되기 때문에 여전히 안정되지않고 기울어지는 상황이 있었다. 그렇기 때문에 이는 2차 목표를 삼아 차후에 세부조정하기로 결정한 뒤 좌우로만 시점 이동을 가능하게 구현하였다.</p> <p>또한 역시 원손을 오므린 상태에서 오른손의 검지만 앞으로 펴게 되면 카메라와 립모션센서의 위치가 앞으로 이동하고 검지와 중지만 펴게 되면 뒤로 이동하고 엄지는 좌, 새끼손가락은 우로 이동하도록 코드를 작성하여 구현하였다.</p> <p>2. Object의 크기를 모션으로 제어 및 조정하는 기능 구상 및 코드 제작</p> <p>- 위의 과제를 수행하고 시간이 약간 남았기 때문에 추가적인 기능을 구현할 수 있는 코드를 구상하였다. 우선 코드를 구현하여 보고 테스트를 진행해보기 위해 Sample Object를 생성하고 양 손을 동시에 움켜쥐었을 때 이 두 손의 좌표를 구하여 Frame마다의 거리차이를 계산하고 이를 기존 Object의 크기의 x,y,z에 적용하는 것으로 구상을 하였다. 이렇게 진행하였을 때 예상되는 문제점은 손의 위치에 따라 상당히 민감하게 작용될 것인데 우선 기능 구현이 되는지가 1차 목표이기에 구현 후 문제가 발생한다면 2차 진행때 수정할 생각이다.</p>		

개발일지			
날짜	10/2/2018	작성시간	18:42:00 PM
이름	조덕진		
개발과제	1. Object의 크기 제어 및 조정 코드 구현 2. Sample UI를 이용하여 이를 카메라에 3D 오브젝트화 수행		
내용	<p>1. Object의 크기 제어 및 조정 코드 구현</p> <p>- Leap Motion을 이용하여 해당 Object의 크기를 제어 및 조정을 하여야 하기 때문에 Sample Object를 생성하여 이를 가지고 테스트 및 프로젝트를 진행했다. 크기의 조정은 양손의 거리차이를 이용하여 변경하는 방식으로 하였다. 제스처는 양손을 움켜쥐었을 때이며 이전 프레임에서 양 손바닥의 거리를 구하고 다음 프레임에서 또 다시 양 손바닥의 거리를 구한다. 이 두 프레임 사이에 구한 손바닥사이의 거리가 다르다면 이것이 크기조정에 필요한 수치이기 때문에 차이 값을 현재 작업하고 있는 Object의 기존 크기값에 더해주었다. 이러한 방식으로 Object Scale의 x,y,z를 변경하여 크기를 조정/제어 하였다.</p> <p>진행 중 오류 사항으로는 이 크기 제어 기능은 프레임사이 마다의 차이를 이용한 것이므로 상당히 민감하게 작동하고 크기변화도 심하였다. 그렇기 때문에 변경해주는 수치의 값에 매우 작은 값을 곱하여 비례감소시켜 적용시켜 민감도를 낮췄다.</p> <p>2. Sample UI를 이용하여 이를 카메라에 3D 오브젝트화 수행</p> <p>- 임시의 2D UI를 제작하여 이를 림모션을 이용하여 터치할 수 있는 가를 테스트하는 작업이었다. 하지만 Leap Motion의 적외선 센서가 손을 인식하는 거리가 제한되어 있고 2D UI는 화면에 고정적으로 있는 것이기 때문에 손으로 인식할 수 없는 거리에 있었다. 그렇기 때문에 UI가 2D가 아닌 3D 오브젝트화 시켜 이를 Scale을 감소시켜 Main Camera의 자식으로 두면서 해결을 하였다.</p>		

개발일지			
날짜	10/4/2018	작성시간	19:11:00 PM
이름	조덕진		
개발과제	1. Leap Motion으로 손 인식을 이용한 클릭 이벤트와 기존 기능 병합 2. 작업된 Unity UI와 1차적으로 코드 병합		
내용	1. Leap Motion으로 손 인식을 이용한 클릭 이벤트와 기존 기능 병합 - 다른 팀원 구현한 Leap Motion의 손 인식을 Main Controller로써 마우스를 대체하여 클릭 이벤트를 발생시키는 기능을 기존의 구현된 기능들과 병합을 시켰다. 손동작을 인식해 UI창을 띄우며 카메라의 좌우이동, Object를 손을 이용하여 터치하였을 때 해당 Object를 선택 및 크기 조정/제어 그리고 Grab하여 위치 이동, Camera의 위치조정 기능들을 하나의 프로젝트에 병합하였다. 2. 작업된 Unity UI와 1차적으로 코드 병합 - 위에 언급한 Gesture을 이용한 구현된 기능코드들 과 Unity UI를 전부 병합하여 작동되는지 테스트진행하였다. 손바닥의 회전에 따른 카테고리UI를 Enable 시키고 손 모션으로 3D 오브젝트화 된 UI창을 클릭이벤트를 발생시켜 선택을 하며 카테고리별 포함되어있는 Object를 생성, Object를 선택하여 크기 조절 및 위치 이동, Item_Menu를 이용한 회전과 해당 Object 삭제 기능을 테스트를 진행하여 정상 작동이 되는 것을 확인하였다. 또한 시뮬레이션 제작후 동적요소 작동을 위한 On/Off UI도 추가하여 On을 눌렀을 당시 동적요소 작동을 실행시킨 것을 의미하므로 다른 Object들의 편집 및 이동을 방지하고 모든 UI창이 disable 되는 것을 확인하였다. 전체적 기능들을 작동시키며 발생하는 오차범위를 줄여가며 최적화시켰다.		

개발일지			
날짜	10/5/2018	작성시간	18:05:00 PM
이름	조덕진		
개발과제	구현된 UI와 Leap Motion기능들을 병합 및 기능별 코드 분할작업 손 떨림을 인한 UI창의 떨림효과 최소화 작업		
내용	<p><구현된 UI와 Leap Motion기능들을 병합 및 기능별 코드 분할작업/ 손 떨림을 인한 UI창의 떨림효과 최소화 작업></p> <p>- 기존 Unity UI창들과 기능별 코드를 병합하고 필요한 기능들을 테스트 하였다. 병합하는 과정에서 ON/OFF 기능을 추가하여 작동하였을 때 해당 Object에 대한 Item_menu Canvas가 출력되지 않아서 Object를 public으로 대체하여 직접 대입하는 방식으로 해결하였다.</p> <p>또한 손 인식으로 카테고리 UI창을 출력할 때에 Palm.position에 의해 UI의 transform이 결정되도록 구성이 되어있었기 때문에 이를 소수점자리를 조정하는 방식으로 오차를 줄여 떨림현상을 줄이는 방법으로 해결하였다.</p> <p>다른 테스트로는 여러 Object들 중 손 모션으로 touch 하는 방식으로 해당 Object를 선택하여 크기 및 위치 조정, 제어하고 Item_menu UI창을 띄워 회전 또는 Object 삭제기능을 테스트하여 정상작동하는지 확인하였다.</p> <p>UI창의 카테고리별 Inventory UI창에 Sample Object들을 임의로 DataBase에 추가를 시켜 전체적으로 구현한 모든 기능이 정상 작동하는지 테스트를 하였다.</p>		

개발일지			
날짜	10/8/2018	작성시간	19:45:00 PM
이름	조덕진		
개발과제	1. 특정 AI의 움직임에 대한 코드 구상과 조건 추가(건물붕괴 및 지진)		
내용	<p>1. 특정 AI의 움직임에 대한 코드 구상과 조건 추가(건물붕괴 및 지진)</p> <p>- 통합 작업을 수행하기 위해서 Date를 팀원으로부터 전달받아야 했다. 이 또한 작업 중이었기 때문에 나는 기다리면서 2순위 작업 중 시뮬레이션 내 특정 AI 행동에 대한 상황 구상을 하였다. 구체적으로는 재난상황이며 그 중 건물이 붕괴되는 상황에 대한 시뮬레이션 및 건물의 움직임에 대한 AI 행동 및 전체적 Object의 움직임에 대한 코드/알고리즘을 구상하고 구현을 하기 시작했다.</p> <p>또한 동시에 천재지변 중 하나인 지진이 발생한 상황도 구상하며 이에 대한 AI의 행동 및 알고리즘을 구상하였다.</p>		

개발일지			
날짜	10/10/2018	작성시간	19:00:00 PM
이름	조덕진		
개발과제	<p>1. 시뮬레이션에 필요한 Object Date 추가 수집/ 업데이트 및 분류</p> <p>2. 1차적으로 Leap Motion과 통합된 Sample 프로젝트와 수집된 Object들을 DataBase에 추가후 통합작업</p>		
내용	<p>1. 시뮬레이션에 필요한 Object Date 추가 수집/ 업데이트</p> <p>- 다른 팀원의 작업부분인 Object Date를 수집하고 업데이트를 하는 역할을 내가 역할을 넘겨 받았다. 그렇기 때문에 필요한 Object를 수집하고 부족한 Data 를 추가적으로 업데이트 하였고 이 각각의 데이터들에 스크립트를 삽입하여 전체적으로 사용할 수 있도록 입력하는 작업을 하였다.</p> <p>또한 Object를 카테고리별로 분류하는 작업을 하고 Tag명을 삽입하는 형식으로 구분을 하였다.</p> <p>2. 1차적으로 Leap Motion과 통합된 Sample 프로젝트와 수집된 Object들을 DataBase에 추가후 통합작업</p> <p>- 나의 새로운 작업 파트인 Object 수집과 DataBase Object리스트에 추가하기 위한 작업이 끝난 후 데이터들을 병합하였다. 이를 여태까지 1차적으로 Leap Motion과 UI들을 통합시킨 Sample Project에 결합시켜 해당 인벤토리 UI창과 이를 연동시켜 병합하는 작업을 수행하였다.</p> <p>또한 이 Object들을 대상으로 구현된 기능들을 하나씩 실험해보며 오류사항을 찾고 최적화하는 과정을 수행하였다. 예를 들면 해당 Object를 생성하고 이동시킬 때 RigidBody Component의 미적용으로 인해 작동되지 않았던 것, Tag명을 이용한 Object를 찾는 기능 중 잘못된 Tag Name으로 인한 오류 등 작동시켜보며 이를 찾았고 개선하였다.</p>		

개발일지			
날짜	10/11/2018	작성시간	19:25:00 PM
이름	조덕진		
개발과제	1. 수집한 Object들을 Sample 프로젝트와 연동을 위한 수정작업 및 코드 추가 작업 2. 병합작업에서 생긴 오류 수정 및 전반적 기능 테스트		
내용	1. 수집한 Object들을 Sample 프로젝트와 연동을 위한 수정작업 및 코드 추가 작업 - 수집한 Object를 Sample Leap Motion 프로젝트에 병합시키고 각 기능을 위해서는 해당 객체마다 Collider와 Rigidbody가 필요하다. 그렇기때문에 추가해주며 Collider가 존재하지 않는 Object는 직접 만들어주었다. 또한 Object들을 UI 인벤토리 DataBase에 추가하는 작업을 위해 필요 스크립트를 추가하여 스크립트에 UI 인벤토리 표시를 위한 이미지 추가/ 설명 및 수치 추가를 하였다. 2. 병합작업에서 생긴 오류 수정 및 전반적 기능 테스트 - 위에서 추가한 작업과 업데이트한 DataBase를 추가 적용하여 제대로 적용되어 생성되는 것을 테스트 해보았고 부족하고 더 필요한 Data들은 2차 작업이 제대로 수행되면 필요에 따라 추가할 예정이다. 3. 필요한 3가지 배경 Terrain 구상 및 제작 시작 - 다른 팀원의 역할을 개인사정으로 인하여 내가 맡게 되었으므로 전체적인 시뮬레이션 제작에 필요한 Terrain을 3가지정도 제작할 계획을 회의를 통하여 결정하였으며 이에 각 Sample에 대한 크기/ 지형을 구상하고 시험제작을 하였다.		

개발일지			
날짜	10/12/2018	작성시간	18:55:00 PM
이름	조덕진		
개발과제	1. 1차 Sample Project 테스트후 오류 사항 개선방안 탐색 및 구상 (Object의 이동)		
내용	<p>1. 1차 Sample Project 테스트후 오류 사항 개선방안 탐색 및 구상 (Object의 이동)</p> <p>- 1차 우선순위 목표를 위한 기능의 구현이 완료되었다. 그렇기 때문에 이를 전체적으로 Test하는 작업을 진행중이다. 하지만 시뮬레이션 구동시에 발생하는 문제점들과 오류사항이 상당히 많았으며 이를 개선하는 작업이 꽤 오래걸릴 것 같다.</p> <p>구체적인 개선사항으로 Object를 처음 생성시 어느 위치에 생성할지 미리 보여주는 것을 개선해야하고 생성된 Object를 이동시키는 방법이 손으로 집어서 움직이는 형식이었다. 실제 Test를 해보니깐 상당히 비효율적이었다. 그래서 우선적으로 움직이는 방식을 손으로 직접 집는 방법 대신 RayCast를 이용하여 원거리에서 이동시키는 방식으로 대체하였다. 이 방법 역시 세밀한 조정이 불가능하여 이를 최적화하고 효율적으로 작동시키기위한 작업을 하였다. 또한 Object의 이동할 시에 Object의 표면을 기준으로 지면에 붙어서 Slide되는 개선방안을 결정하고 스크립트를 구현하기 위한 탐색 및 구상을 하였다.</p>		

개발일지			
날짜	10/15/2018	작성시간	18:55:00 PM
이름	조덕진		
개발과제	1. 1차 Sample Project 테스트후 오류 사항 개선방안 탐색 및 구상 (Object의 이동)		
내용	<p>1. 1차 Sample Project 테스트후 오류 사항 개선방안 탐색 및 구상 (Object의 이동)</p> <p>- Object를 이동시킬때 LeapMotion을 이용하여 생성된 손 모델의 Index손가락에서 Raycast가 나오도록 설정하였다. 이 Raycast가 특정 Layer을 가진 물체에 닿았을 때 그 좌표를 구하여 해당 Object의 위치를 좌표값으로 이동시키는 방법으로 바닥에 붙어 움직일 수 있도록 구현하였다. 또한 Object를 이동시키는 와중에는 Rigidbody의 Iskinematic 기능을 On 하고 충돌 기능을 OFF하여 움직여서 먼저 생성된 Object들의 위치에 영향을 끼치지 않도록 하였고 이동을 종료하였을 때 기능들을 원상복구 시켜 고정되게 진행하였다.</p>		

개발일지			
날짜	10/16/2018	작성시간	19:35:00 PM
이름	조덕진		
개발과제	1차 발표 준비 작업		
내용	<p>1. 1차 발표 준비 작업</p> <p>- LeapMotion을 이용한 Object배치 및 이동 수정작업 마무리 작업을 하였다. Leap motion의 기술적 한계로 인해 Object 배치와 이동하는 기능이 정밀하게 의도대로 수행되지 않는 문제점이 있어 이를 수정하는 작업을 해왔지만 기술적 문제인 만큼 최대한 오차 값을 수정하고 프레임의 시간 차이도 늦춰보고 좌표값을 정수화하여 맞춰 표준화하는 노력을 하였다.</p> <p>2. 1차 발표를 위한 Project 구성, ppt제작, 내용정리</p> <p>- 1차 발표를 위해 현재까지 진행된 상황을 정리하고 문서화 하였으며 이를 ppt로 만들어 정리를 하였다. 또한 발표에 필요한 기능설명 등을 동영상으로 촬영하여 설명하기 위해 동영상촬영을 하였고, Project 발표시 시연을 해야하는데 이 때 안되는 상황에 대비하여 동영상촬영 역시 하였다.</p>		

개발일지			
날짜	10/17/2018	작성시간	18:00:00 PM
이름	조덕진		
개발과제	프로젝트 1차 발표		
내용	<p><프로젝트 1차 발표></p> <p>- 우리 팀의 1차 발표의 목적은 UI창 구현, 그리고 Leap Motion을 Main Controller로써 대체하여 이를 이용한 모든 기능을 컨트롤 할수 있게 하는 것이었다. 그리고 Inventory UI창의 Object Data List에 사전에 특정 스크립트가 추가되고 크기조정 및 RigidBody 추가등 작업이된 Data들을 등록시켜 이 Object들을 이용하여 실험할 수 있으며 배치/이동 시키고 회전, 삭제가 가능하게 하는 것이었다.</p> <p>진행상황으로는 목표했던 모든 기능은 구현이 가능하였고 1차 발표를 하였다. 하지만 막상 프로젝트가 진행되고 시연까지 해보았을 때 생각보다 많은 오차가 발생하였다. Leap Motion 기기의 한계/ 기술적 한계 때문에 정확한 시연이 불가능 하였고, 이 때문에 시뮬레이션 구동시 Object의 배치/이동이 정밀히 되지 않으며 손모션또한 인식을 정확히 하지 못하였다.</p> <p>이러한 문제를 가지고 팀원들과 회의를 거친 결과 Main Controller는 다시 키보드/마우스로 사용하기로 하였으며 홀로그램에 중점을 두는 방향으로 목표를 잡았다. 그리고 Leap Motion은 특정한 파트 혹은 세밀한 시뮬레이션 작업이 필요하지 않는 기능에서 사용하기로 결정을 하였다.</p>		

개발일지			
날짜	10/18/2018	작성시간	20:30:00 PM
이름	조덕진		
개발과제	1. Main Controller를 LeapMotion에서 키보드/마우스로 변환 작업		
내용	<p>1. Main Controller를 LeapMotion에서 키보드/마우스로 변환 작업</p> <p>- 우리팀은 1차 발표 후 목표의 방향을 바꾸었다. Main Controller을 Leap Motion이 아닌 키보드/ 마우스로 대체하기로 하였다. 그렇기 때문에 이 작업을 진행하였다. 기존의 3D object화 되어있던 UI창들과 포함되어진 기능들을 다시 2D화 시켰고 Event 발생도 역시 마우스 버튼클릭으로 수정을 하였다. Object 배치 및 이동 시에 기존에는 Leap Motion에서의 손가락에서 Raycast를 이용하여 선택 및 설치/이동을 시켰었지만 마우스 커서의 position을 이용하여 좌표를 구하고 이를 통해 구현하는 방식으로 대체시켰다.</p> <p>2. 동적 요소 변환요소 구현 및 UI추가</p> <p>- 동적 변환요소, 즉 현재까지 임시적으로 구현된 날씨변화, 재난상황에 해당하는 요소들에 대하여 각각 UI창을 만들었으며 기존 진행중이며 병합시킨 Project에 이것들 또한 병합시키는 작업을 진행하였다.</p>		

개발일지			
날짜	10/19/2018	작성시간	19:10:00 PM
이름	조덕진		
개발과제	1. LeapMotion에서 키보드/마우스로 변환 작업 마무리 2. Object List내의 Object들의 수정작업		
내용	<p>1. LeapMotion에서 키보드/마우스로 변환 작업 마무리</p> <p>- 어제하던 작업에 이어 변환 작업을 마무리 하였다. 진행된 사항으로는 마우스와 키보드를 이용해 전체 시뮬레이션을 제어하다보니 움직임이 좀더 자유로워졌다. 하지만 이에 동반된 문제점이 마우스 position을 이용하여 Raycast를 보내서 Object들을 설치/이동시켰는데 거리가 짧아 선택이 되지 않았었다. 그렇기에 Raycast의 거리를 비약적으로 증가시켜 해결하였다. 기존의 Leapmotion으로 개발했던 것들을 Main 입력을 바꾸고 나서 모든 기능이 구현되고 작동되는 것을 확인하였다.</p> <p>2. Object List내의 Object들의 수정작업</p> <p>- Object List 내의 Object들의 크기조정 및 Renderer 및 Collider를 수정하였다. Object들을 설치할 때 문제점이 발생하였다. 앞서 설명하였듯 Object를 배치 및 설치할 때 마우스 Position의 좌표값을 구하여 여기에 배치를 하였는데 Object들 마다의 Position은 객체의 정중앙 좌표로 지정되기 때문에 마우스 좌표에 이동시켰을 경우 바닥을 파고들어가는 문제가 발생하였다. 이를 해결하기위해 Object들 마다 Mess Collider을 추가한뒤 선택된 Object의 Renderer을 구하여 이의 높이를 구하고 이 높이 만큼 위로 이동시켜 생성해주도록 하였다.</p> <p>또한 Road에 해당하는 Object들은 prefab 설계 단계에서부터 상대좌표값이 틀어져 있기 때문에 설치할 때 회전을 시켜 생성하는 방향으로 해결하였고, Object들 회전시킬 때에는 각 Object마다 상대좌표에 의해 회전되었기 때문에 이를 절대좌표 기준으로 회전되게 수정하였다. 추가 문제점은 추후에 개선해 나갈 것이다.</p> <p>3. 동적 상황변환에 대한 Object 파티클 추가</p> <p>- 우선적으로는 Light요소를 넣어주며 밤낮 변화와 날씨 변화에 의해 On/Off 되는 것을 시각적으로 표현하여 이펙트를 주었고 다른 시각적인 부분은 팀원들과 회의를 통해</p>		

개발일지			
날짜	10/22/2018	작성시간	18:00:00 PM
이름	조덕진		
개발과제	1. 현재까지 진행된 UI, Cotroll요소 와 기본적인 AI움직임 병합과정		
내용	<p>1. 현재까지 진행된 UI, Cotroll요소 와 기본적인 AI움직임 병합과정</p> <p>- 전달 받은 Prefab과 코드들을 하나의 프로젝트로 병합과정을 진행하였다. 팀원들이 수정한 카테고리별 Object리스트를 전달받아 이를 현 통합적으로 병합하면서 진행하는 프로젝트에 결함시키고 이에 해당하는 기능들을 하나씩 구현해보며 오류를 찾고 이를 수정하는 작업을 하였다. AI의 움직임을 위한 Navigation 기능을 시현해 보면서 필요한 요소들을 추가하였고, Vehicle에 해당하는 Object들이 선택되지 않은 오류가 발생하여 원인을 Shader에서 찾고 이를 수정하여 해결하였다.</p> <p>또한 작업간 틈이 생기는 시간을 이용하여 전체 Background에 해당하는 예제 Terrain을 도시, 산 지형을 의도로 2개 제작을 하였다.</p> <p>- 앞으로 구현해야할 동적 환경요소 변화에 대한 AI 대응 움직임들 등을 팀원들과 구상해보고 대략적인 회의를 하며 분류작업을 하였다. 현재 진행중인 상황변화 요소들을 마무리하게 되면 본격적인 동적 요소에 대하여 Object들, AI의 움직임을 구현하는 작업에 들어갈 것이다.</p>		

개발일지			
날짜	10/23/2018	작성시간	18:00:00 PM
이름	조덕진		
개발과제	1. AI 동적요소 상황변화(건물붕괴)를 화재로 변경작업		
내용	<p>1. AI 동적요소 상황변화(건물붕괴)를 화재로 변경작업</p> <p>- 동적 상황변화들 중에 카테고리를 살펴보면 건물붕괴, 지진, 태풍이 있었다. 이 항목들은 임시적으로 생성해 놓은 것이기 때문에 팀원들과 상의를 한 결과 건물붕괴는 지진에 포함된 개념이므로 다른 것을 제작하기로 결정하였다. 그래서 나는 건물붕괴보다는 건물에서의 화재를 발생시키는 것으로 변경하였다.</p> <p>'건물화재' 카테고리를 선택하게 되면 Main Camera기준으로 중앙지점에서 Raycast를 발사하게 된다. 그리고 마우스 클릭을 하게된다면 마우스 포지션으로 Raycast가 지나가게되며 이게 Object와 부딪혔을 때 부딪힌 대상이 건물Object이면 해당 건물이 화재의 건물로 선택에 되어지는 원리로 구현을 하였다. 선택된 Object의 Position에 화재 파티클을 생성시키고 다른 카테고리가 선택되기전까지 이를 지속시키며 화재 상황을 구현하였다.</p>		

개발일지			
날짜	10/24/2018	작성시간	20:10:00 PM
이름	조덕진		
개발과제	1. 상황 변화(화재)에 따른 추가적인 AI Object의 생성 및 배치		
내용	<p>1. 상황 변화(화재)에 따른 추가적인 AI Object의 생성 및 배치</p> <p>- 상황변화 건물 화재 선택 시 추가적으로 필요한 Object의 생성할 필요가 있었다. 단순한 '건물붕괴' 상황을 '건물화재'로 바꾸면서 이에 상응하여 변하는 동적요소로는 '소방차', '응급차' 등 추가적인 Object 생성이 필요했다.</p> <p>UI창에서의 해당상황이 클릭되었을 때 시뮬레이션을 작동하며 설치된 Road 태그를 가진 모든 Street를 읽어 랜덤값으로 생성되게 하였으며 생성된 Object들은 화재상황으로 선택된 건물을 감지하고 읽어 들였던 모든 Street들과 건물의 position의 거리를 비교하고 가장 가까운 곳을 목표로 하여 이동하게 구현을 하였다. 스크립트를 작성하고 구현하는 도중 AI 자동차의 움직임 구현이 아직 미완성이었으므로 테스트를 정확히 하기에는 지장이 있었다. 그렇기 때문에 우선적으로 Object들 생성과 각 Object들이 이동할 Target 위치를 확인하는 것으로 Test를 하였으며 이 후 AI 자동차의 움직임 구현이 완료되었을 때 이를 병합하여 이어서 테스트를 할 계획이다.</p>		

개발일지			
날짜	10/25/2018	작성시간	18:27:00 PM
이름	조덕진		
개발과제	1. 상황 변화(화재)에 따른 AI Object(사람)의 움직임 구상 및 구현		
내용	<p>1. 상황 변화(화재)에 따른 AI Object(사람)의 움직임 구상 및 구현</p> <p>- 상황변화 건물 화재 선택 시 동적 요소 AI Object(사람)의 움직임을 구현하였다. 일반적으로 화재가 건물에 화재가 났을 때 사람의 행동마다 다르겠지만 안전을 우선적으로 생각하여 시뮬레이션 구동 시 설치하여 자동적으로 움직이고 있는 AI들의 움직임을 제어하였다.</p> <p>먼저 AI 사람 Object의 움직임을 간략히 말하자면 시뮬레이션 구동 시 설치된 건물들을 임의로 목적지로 설정하여 이동하고 도착 후 목적지를 재설정하여 움직이는 방식이다. 이를 수정하여 UI창의 '건물 화재' 버튼이 클릭되어 이벤트가 발생하고 특정 건물을 또 다시 선택하여 화재 상황을 부여하였을 때 모든 AI 사람 Object들의 목표 목적지 목록에서 이 건물을 제외시키며 다가가지 못하게 설정을 하였다. 발생된 예외상황으로 화재건물 뒤쪽을 목표로 하여 가는 경우 이 지역을 관통하여 가는 상황이 있었으며 이 또한 화재건물의 주변의 Obstacle을 2배로 설정하여 AI Object 들이 우회하게끔수정/ 구현하였다.</p>		

개발일지			
날짜	10/26/2018	작성시간	20:55:00 PM
이름	조덕진		
개발과제	1. 상황 변화(화재)에 따른 AI Object(사람)의 움직임 구현완료 2. 상황 변화(화재)에 따른 추가적인 AI Object의 생성 및 배치 완료 및 병합		
내용	1. 상황 변화(화재)에 따른 AI Object(사람)의 움직임 구현완료 - 다른 팀원으로 부터 수정된 AI 자동차 Object의 움직임 기능을 전달받아 추가적인 AI Object 생성 및 배치 기능의 구현을 완료하였으며 그저께와 어제 구현한 상황변화(화재)에 따른 AI Object(사람)의 움직임 및 추가적인 AI Object 생성 기능을 하나로 합쳐 구현, 해당 상황변화요소의 구동확인을 마쳤다. 추가적으로 생성된 Object 들에 상황에 맞는 particle effect를 주어 조금 더 시각적인 효과를 증가시켰다. 2. 상황 변화(화재)에 따른 추가적인 AI Object의 생성 및 배치 완료 및 병합 - 팀원들이 작업하여 구현 완료된 다른 상황변화(태풍, 지진)에 대해 구현된 기능들을 하나의 Project의 파일로 병합을 하였고 이 과정에서 발생하는 시행오류를 검출하고 수정하였으며 작성된 Script를 하나의 형식으로 통일 시키는 작업을 진행하였다. 통합시키는 작업은 완료가 되어 각 기능은 정상적으로 작동하지만, 전체적으로 모든 기능들에 사소한 오류 및 예외상황이 발생하여 이를 처리하고 수정하는 작업이 필요하다고 느껴서 이 문제는 추후에 처리하기로 하였다.		

개발일지			
날짜	10/29/2018	작성시간	19:06:00 PM
이름	조덕진		
개발과제	1. 현재까지 구현된 기능과 동적요소를 전부 병합 및 최적화 2. 추가 구현할 기능 또는 변화요소를 협의 및 토론		
내용	1. 현재까지 구현된 기능과 동적요소를 전부 병합 및 최적화 - 현재까지 계속 각 동적변화요소를 추가하고 수정 / 최적화 할때마다 변화된 것들을 병합하였다. 업무를 분업화하여 프로젝트를 진행하는 것이 효율적일 수 있으나 이 경우에는 각자 분업화하였기 때문에 개인마다 구현한 기능을 병합시키는데 어려움이 있을 것이며 시간도 많이 걸릴 것으로 판단하였다. 그래서 분업화 하되 주기적으로 프로젝트를 병합시키는 과정을 하였다. 오늘은 병합된 동적요소들을 전부 테스트하며 이에 발생하는 오류검출과 예외처리 및 수정 작업을 진행하였다. 구체적으로는 동적 상황변화에 따라 생성되어 시뮬레이션이 되는 Object들의 위치가 복원되지 않았던 오류, 화재상황 시 On/Off 기능에 영향을 받지 않아 계속 화재상황이 유지되던 오류들을 수정하였다. 2. 추가 구현할 기능 또는 변화요소를 협의 및 토론 - 추가적으로 구현할 기능들에 대해 토의 및 분업화 하였고 2차 우선순위 작업과 최적화 작업완료가 계획한 시기보다 앞당겨진다면 미리 추가적으로 진행하게 될 3차 우선순위작업에 대해 구상 및 토의를 하였다.		

개발일지			
날짜	10/30/2018	작성시간	20:55:00 PM
이름	조덕진		
개발과제	1. AI Navi 병합과정 중 생긴 특정 Object생성시 발생하는 오류 수정 및 전체적 최적화 작업들을 하나의 Project로 병합		
내용	<p>1. AI Navi 병합과정 중 생긴 특정 Object생성시 발생하는 오류 수정 및 전체적 최적화 작업들을 하나의 Project로 병합</p> <p>- 우선적으로는 곧 2차 발표가 있기 때문에 최대한의 오류검출 및 수정, 그리고 최적화 작업이 선행되어야 된다고 팀 회의를 통해 결정하였다. 다른 팀원들이 어제 수정하고 기능을 추가시켰던 작업을 오늘 하나의 Main Project에 병합시키는 과정을 진행하였다. 진행하며 발생하는 오류를 수정해가며 합쳤으며 AI Navigation 기능 중에 경로를 표시해주는 Bake 기능이 특정 Object 생성 시 오류를발생시켰다.</p> <p>이는 기존에 Bake 과정 중에 특정 Object 높이의 문제로 경로에 혼선을 주웠던 것이었으며 해당 Object를 삭제시키는 방향으로 결정하였다.</p> <p>두 번째로 병합작업 후 시간이 남았으므로 Unity 프로그램 내에서 구동 중에 현재까지의 Scene을 저장시키는 기능을 다른 팀원과 같이 구상하고 구현하였다. 이는 Scene에 존재하는 모든 Object들이 ID, Name/Tag를 갖고 있는데 이를 이용하여 Position, Rotation 값을 문서화하여 저장하고 파일로 만들어 저장하는 방법을 이용하였다.</p>		

개발일지			
날짜	10/31/2018	작성시간	19:02:00 PM
이름	조덕진		
개발과제	1. 전체적 최적화 작업들을 하나의 Project로 병합 2. 구현한 Scene 저장/불러오기 기능 구현 완료		
내용	<p>1. 전체적 최적화 작업들을 하나의 Project로 병합</p> <p>- 우선 현재 하나의 Project로 통합된 기능들을 구현하며 오류를 검출하였다. 어제는 구현되는 기능들이 오늘은 안되는 것들이 있었다. 구체적으로는 AI Navigation이 정상 작동되지 않아 AI 사람 Object가 경로를 찾지 못하는 경우가 발생하였고 건물 Object 들의 기준점 위치가 이상하여 위치 Target을 찾지 못하는 경우도 발생하였다. 이러한 것들은 팀원들과 상의를 한 후 해결방법을 토의하였다. 또한 화재 상황 동적요소 시 추가했던 시각적인 요소를 부과시켜주는 화재건물에 물뿌리는 기능이 방향을 제대로 찾지 못하는 상황이 발생하여 이것을 부모에 상속시켜 부모의 Rotation을 받아와 다시 회전시켜 맞추는 방식으로 해결하였다.</p> <p>이외에도 발생한 사소한 에러로는 시간/ 조도의 자동적으로 변화하고 수동으로 할 수있는 전환 기능 추가하는 부분에서 수동에서 해당 변화에 따라 적용되는 AI Object 요소들이 작동하지 않는 것이 있었는데 이것은 전환 하여 시간/조도를 변화 하는 스크립트에 AI Object가 변화하는 스크립트를 추가하여 수정하였다.</p> <p>2. 구현한 Scene 저장/불러오기 기능 구현 완료</p> <p>- 어제는 Scene을 ID와 Name/Tag를 이용하여 Position, Rotation 값을 읽어 들여와 문서화하여 저장하는 방식으로 기능을 구현하였다. 오늘은 이 문서를 불러와 분석하여 또한 Name/Tag를 이용하여 Object Prefab 을 찾고 문자열을 분석하여 Position, Rotation 값을 적용하여 생성하는 방식으로 불러오기 기능을 구현하였다. 하지만 여기서 생긴 문제점이 저장을 하면 스크립트 내에 지정한 경로의 지정된 이름으로 파일이 생성되는데 저장을 계속 하게 되면 기존의 파일에 덮어쓰는 방식으로 저장되어 가장 최신 저장 파일만 불러올 수 있는 것이었다. 이는 추후 시간, 또는 파일명을 지정할 수 있게하여 분류하고 이것들을 목록으로 불러올 수 있는 방식으로 개선해 나갈 생각이다.</p>		

개발일지			
날짜	11/1/2018	작성시간	19:00:00 PM
이름	조덕진		
개발과제	1. 전체적 최적화 작업들을 하나의 Project로 병합		
내용	<p>1. 전체적 최적화 작업들을 하나의 Project로 병합</p> <p>- 다른 팀원이 기능을 추가시켜 구현한 Save/Load 작업을 현재 Project에 병합하는 과정을 진행하였다.</p> <p>UI를 통합시켜 구동해보고 기존 Project와 결합시켜 정상작동되는 것을 실험을 해보았다. Test 과정에서 Save/Load 하는 경로지정으로 오류가 발생하였지만 절대경로를 사용함으로써 해결하였다. 또한 곧 2차 발표가 있으므로 현재까지의 작업된 Project를 계속 여러가지를 Test하면서 오류를 검출하여 수정하고 Object마다 발생하는 오차를 줄여나가는 작업을 수행하였다.</p> <p>2. 3차 우선순위 작업 중 홀로그램 출력을 위한 작업 및 구상</p> <p>- 다른 팀원의 작업을 전달받기 전까지 시간이 남았었다. 그렇기 때문에 3차 우선순위 작업들 중에 홀로그램 제작 작업이 있다. 이를 수행하기 위한 전반적인 구조 구상과 진행에 대하여 생각을 하고 디자인을 수행하였다. 구상하는 도중 중요한 문제가 발생하였었다. 홀로그램 출력을 위하여 Main Camera를 제외한 4개의 Camera를 두고 이를 4 방면에서 출력을 하게 되는데 모니터 1개에 출력하게되면 4개의 카메라화면을 분할 출력해야하므로 홀로그램의 크기가 매우 작게 출력될 것이라는 점이였다. 또한 이러한 화면이 홀로그램으로 제대로 출력되는지조차도 의문점이 생겨났다. 이러한 문제들은 2차 발표가 끝난후 팀원들과 토의해볼 계획이다.</p>		

개발일지			
날짜	11/2/2018	작성시간	17:48:00 PM
이름	조덕진		
개발과제	1. 2차발표를 위한 Project 최적화 및 발표준비(ppt제작)		
내용	<p>1. 2차발표를 위한 Project 최적화 및 발표준비(ppt제작)</p> <p>- 월요일이 2차 발표 날이다. 그렇기 때문에 주로 시뮬레이션을 구동시켜 점검위주의 작업을 진행하였다. 그에 맞게 1차 발표때와는 수정한 사항과 개발 방향이 바뀐것이 많았기 때문에 PPT 작업 및 발표준비 과정에서 수정이 필요하였다.</p> <p>발표를 대비하여 PPT 구성을 팀원들과 회의를 하여 구상하였고 팀원 수 역시 5명에서 3명으로 줄었기 때문에 발표를 진행할 파트를 다시 나누었다. 주어진 파트에 맞게 발표 준비를 하고 피피티를 작성하였다. 그 후에는 처음에 언급한 것 처럼 시뮬레이션 Test를 계속 진행하였다. 팀에서 계속 Test를 진행하는 이유는 UI제작 같은 기능은 문제가 되지 않지만 AI 인공지능 타입의 움직임, 동적상황요소는 예외의 상황과 오차가 상당히 많이 발생하고 오차/오류가 없던 부분에서 발생하기도 하였고 대부분의 Event 들이 연결되어 있기때문에 최적화 작업이 항상 필요하며 수정이 필요하기 때문이다.</p>		

개발일지			
날짜	11/5/2018	작성시간	19:00:00 PM
이름	조덕진		
개발과제	프로젝트 2차 발표		
내용	<p><프로젝트 2차 발표></p> <p>- 오늘은 프로젝트 2차 발표하는 날이었다. 그렇기 때문에 발표전 까지 작성한 PPT를 기반으로 팀원들과 발표준비를 하였다. 또한 동시에 현재까지 구현한 Project를 Build과정을 하여 발표준비를 하였는데 예상치 못한 해상도 차이 때문에 UI문제가 발생하였고 Build하기전의 개발환경에서 맞춰둔 오차범위 또는 Navigation기능들이 Build를 한 후와 차이가 발생하였다. 발표전까지 이를 수정하며 오차를 줄여나갔지만 개발환경에서 수정한 만큼은 되지 않았다. 이를 통해서 다음 발표준비를 하면서는 최적화작업과 동시에 Build과정을 병행하여 개발환경이 아닌 Build and run 환경에 맞춰서 오차를 줄여나가야하는 것을 느끼고 깨달았다.</p> <p>1차발표때와는 다르게 미리 팀원들과 구체적으로 발표파트구상을 하였기 때문에 발표는 이전보다 점더 성공적으로 하였으며 내일부터는 3차 우선순위 작업을 진행할 예정이다.</p>		

개발일지			
날짜	11/6/2018	작성시간	19:05:00 PM
이름	조덕진		
개발과제	1. PC버전으로 개발된 Project에 일부 기능에 Leap Motion기능을 접목 2. 자체제작 카테고리의 Object 추가 및 설정		
내용	<p>1. PC버전으로 개발된 Project에 일부 기능에 Leap Motion기능을 접목</p> <p>- 1차 우선순위 작업으로 LeapMotion의 손 모션별로 기능을 구현하였었다. 2차작업으로는 Project의 메인 프로그램 건축시뮬레이션을 pc버전으로 개발을 하였다. 이제는 3차 우선순위 작업을 하는 순서이므로 이 2개의 모드를 하나로 합치기로 했다. 1차 우선순위 작업 즉, LeapMotion을 Main Controller로써 시뮬레이션 프로그램을 구동하기에는 기기가 가진 한계점이 너무심하여 불가능 하였다. 그렇기 때문에 Main Controller는 PC의 입출력으로 하되 시뮬레이션을 작동시키는 ON 모드일 때 제한적으로 LeapMotion을 사용하기로 하였으며 ON모드 UI에 LeapMotion모드를 추가하여 전환이 가능하도록 하였다. 또한 UI는 LeapMotion으로 클릭이 불가하기 때문에 림모션 모드일때에는 전용 UI를 3D Object화 하여 특정 모션으로 출력을 하고 해당 UI를 클릭하여 이벤트를 발생시키도록 하였다.</p> <p>2. 자체제작 카테고리의 Object 추가 및 설정</p> <p>- LeapMotion모드를 병합하기 이전에 빠르게 완료 가능한 작업을 우선적으로 하기로 결정하였다. 그렇기 때문에 카테고리UI의 가장 아래칸 '자체제작'파트를 완성시키기로 하였다. Cube Object를 색깔 별로 데이터베이스에 등록하고 시뮬레이션에 필요한 여러 Script 들을 추가하였다. 또한 이것을 장애물 처리를 하여 사람 AI들이 이것을 마주하게 되면 장애물로 인식하여 피해가고 이동하는 루트를 조정하도록 하였다.</p> <p>이 자체제작 파트는 데이터베이스에 미리 등록된 주어진 Object들 이외에 큐브를 쌓아 원하는 것을 만들수도 있고 장애물을 설치할수 있도록 하였다. 흡사 마인크래프트를 모티브로 추가한 파트이다.</p>		

개발일지			
날짜	11/7/2018	작성시간	19:30:00 PM
이름	조덕진		
개발과제	1. 홀로그램 출력을 위한 하드웨어 제작 구상 2. TrinusVR기술을 이용해 프로젝트 VR구현 및 기존 프로젝트와 병합		
내용	1. 홀로그램 출력을 위한 하드웨어 제작 구상 - 이제 홀로그램을 어떻게 만들지 팀원들과 토의를 하고 디자인을 구상하였다. 회의 중에 나온 의견들로는 모니터 하나로 카메라 4개를 작게 출력하는 방식이고 다른 하나는 모니터 4개를 사용해서 각각 카메라 1개씩 출력을 하는 방식이다. 처음 계획으로는 학원에서 사용중인 TV를 모니터로써 빌릴 생각이었으나 문의하였는데 불가능하였다. 그렇기 때문에 위의 2가지 의견이 나왔다. 2가지 의견 모두 장단점이 존재하였다. 모니터 1개를 사용하는 의견은 재료 또는 필요, 추가 비용이 많이 들어가지 않는다는 장점이 있지만 출력이 작게 된다는 단점이 있었다. 반대로 모니터 4개를 사용하는 것은 출력이 모니터 1개 정도의 크기만큼 출력된다는 장점이 있지만 이를 위해서는 추가 구입해야하는 것이 상당히 많았으며 홀로그램 크기도 상당히 커질것으로 예상되었다. 2. TrinusVR기술을 이용해 프로젝트 VR구현 및 기존 프로젝트와 병합 - 어제 PC모드와 LeapMotion 모드의 병합작업을 시작하였었다. 오늘은 어제의 작업을 이어서 이전에 구현 완료되었던 동적상황 변화 요소들이 제대로 작동되는지 확인하였다. LeapMotion 전용 UI도 적용시켜 손으로 특정 모션을 취하였을 때 해당 동적상황 변화 요소들이 출력되며 손가락 클릭을 이벤트 시스템으로 전환하여 이벤트발생이 가능하게 하였고 다시 PC모드로의 전환도 정상적으로 되는 것을 확인하였다. 이 뿐만아니라 Trinus VR을 적용시켜 휴대폰과 연동하여 CardBoard 방식으로 PC모드가 VR로 연동되는 것을 확인하였다.		

개발일지			
날짜	11/8/2018	작성시간	18:20:00 PM
이름	조덕진		
개발과제	1. 홀로그램 출력을 위한 하드웨어 제작 2. 프로젝트 Build시 LeapMotion의 좌표 설정 오류검출		
내용	1. 홀로그램 출력을 위한 하드웨어 제작 - 홀로그램 제작 방식에 대하여 계속적인 토의를 하였다. 그렇게하여 결정된 방식이 모니터1개로 작더라도 구현가능성이 높은 방식을 택하기로 결정하였다. 이를 구현하기 위하여 카메라를 4개를 생성하여 화면을 분할하여 4개로 Display되는 것을 Test해보았다. 2. 프로젝트 Build시 LeapMotion의 좌표 설정 오류검출 - 어제 LeapMotion을 PC모드에 병합시키고 정상작동되는 것을 확인하였다. 구현했던 기능들도 정상작동 되었으며 오류는 검출되지 않았었다. 하지만 오늘 통합된 Project를 Build 하여 팀원들에게 공유하는 과정에서 뜻하지 않은 오류를 검출할 수 있었다. Unity 프로그램 내에서는 구동에는 아무런 이상이 없지만 Build만 하면 LeapMotion UI창의 좌표 오류가 발생하였다. UI는 정상적으로 출력이되는데 클릭 이벤트 좌표가 오류로 발생하였다. Unity내에서 제작할 당시에는 문제가 없고 예러가 없기 때문에 원인을 찾는 작업을 시작하였다. UI창의 이상으로 예상하여 UI를 3D Object화 하기위해 World Space로 맞춰놓았었는데 이를 카메라 스크린에 맞추기도하고 고정좌표로도 변경하였지만 Build 하고 났을 때는 그대로의 오류가 발생하여 원인을 찾지 못하였다.		

개발일지			
날짜	11/9/2018	작성시간	17:55:00 PM
이름	조덕진		
개발과제	1. 홀로그램 출력을 위한 하드웨어 제작 시작 2. 프로젝트 Build시 LeapMotion의 좌표 설정 오류 수정 및 최적화		
내용	1. 홀로그램 출력을 위한 하드웨어 제작 시작 - 홀로그램을 제작에 들어가기에 앞서서 계획을 세우기 시작하였다. 우선적으로 제작에 필요한 재료를 구입할 수 있는 것이 중점이었다. 그렇기 때문에 근처 상점에 가서 PCV 필름을 1개 구입하여 왔고 이를 이용해 모니터의 크기에 맞는지 측정을 하고 앞으로 필요한 작업 또는 추가적으로 필요할 재료를 정리하였다. 본격적인 제작은 월요일부터 시작을 할 것이다. 사전에 Test 해보았던 4개의 카메라 출력이 정상적으로 홀로그램에 비춰질지는 시험적으로라도 제작을 해보고 확인을 할 수 있기 때문에 빠른 제작이 필요로 하여졌다. 2. 프로젝트 Build시 LeapMotion의 좌표 설정 오류 수정 및 최적화 - 어제 Build시 LeapMotion 클릭 이벤트 좌표에 생겼던 오류를 수정하지 못하였다. 어제에 이어 오늘도 거의 하루종일 원인을 찾기 위해 노력을 하였다. 인터넷 검색, 책 검토 등등 여러가지 해보았지만 끝내 자료를 찾지 못하였다. 그렇기 때문에 Collider을 이용하여 클릭이벤트를 대체할 수 있는지도 Test 해보았지만 이 또한 좌표에 오류가 생겨 불가능 하였다. Build를 하지 않으면 오류가 발생하지 않고 매우 정상적으로 작동이 하기 때문에 오류원인 검출이 상당히 어려운 작업이었다. 결국 강사님과 상담한 결과 Build 시 생기는 오류는 Unity내의 자체적인 결점으로 결정짓고 이를 Build 하지 않는 것으로 결정하였다.		

개발일지			
날짜	11/12/2018	작성시간	19:00:00 PM
이름	조덕진		
개발과제	1. 홀로그램 제작을 위한 재료 수집 및 제작		
내용	<p>1. 홀로그램 제작을 위한 구조 설계 및 구성 재료 정리</p> <p>- 4가지방향에서 모두 확인할 수 있는 홀로그램 제작을 하기로 하였다. 그렇기 위해서 전제 제작을 위해 구조를 설계하였다. 기존의 홀로그램 틀이 있었기에 크기를 측정하고 모니터의 크기와 맞는지 비교를 하였다. 그 후 4분면 출력을 위해서는 프리즘 모양의 홀로그램 틀이 필요하며 이를 아크릴 판을 이용하여 제작하기로 결정하였다. 제작을 위해 Sample로써 우선적으로 아크릴판 1개를 구입하여 크기를 대보며 크기를 맞추기위해서 총 4장의 아크릴 판이 필요할 것이었으며 고정하기 위하여 테이프가 필요할 것으로 예상하였다.</p> <p>2. 홀로그램을 위한 영상 출력 및 카메라 구도 조정</p> <p>- 막상 제작을 완료하였다고 가정을 하고 출력이 제대로 될 것인가 에 대한 의문점이 생겼다. 그래서 팀원과 함께 온라인 웹사이트에 포스팅 되어있는 4분면 홀로그램영상을 출력하여 아크릴판에 비춰보며 제대로 출력이 되는지 확인을 하였고 우리의 project의 메인 출력화면 이외에 4개의 카메라를 홀로그램 출력을 위해서만 추가적으로 배치하여 구도를 조정하였다.</p>		

개발일지			
날짜	11/13/2018	작성시간	17:55:00 PM
이름	조덕진		
개발과제	1. 홀로그램 제작을 위한 재료 수집 및 제작		
내용	<p>1. 홀로그램 제작을 위한 재료 수집 및 제작</p> <p>- 오늘은 어제 구조를 설계한 것을 바탕으로 재료를 구입하고 본격적으로 제작에 들어갔다. 피타고라스 정의를 이용하여 홀로그램 틀의 크기를 조정, 이 크기를 바탕으로 아크릴판으로 4분면 홀로그램 틀을 제작하였다. 이를 준비된 틀에 넣고 1차적인 Sample 홀로그램을 완성을 하였다. 이를 Test하여보니 생각지 못한 오류가 검출되었다. 정확히는 오류보다는 오차이었다. 주어진 자원의 한계로 인하여 모니터1개를 4분하여 영상 4개를 출력하다보니 영상의 크기가 다소 작았다. 그렇기 때문에 제작한 홀로그램에 출력은 되었지만 출력한 영상이 작고 출력된 영상 사이사이에 공간이 적다보니 홀로그램틀의 상단부에만 영상이 비춰 전체적으로 영상이 위쪽으로 치우치는 현상이 발생하였다. 홀로그램 틀을 그대로 두는 방향으로 영상의 크기 또는 위치를 조정하여 홀로그램틀의 중앙부에 출력하려고 조정을 해보았지만 이는 모니터의 크기가 작아 생기는 현상이므로 해결하기 위해서는 영상의 크기를 더욱더 작게 만들거나 보다 큰 모니터를 사용하는 방법밖에 없었다. 두가지 해결방법 모두 적용하기 힘들었기 때문에 홀로그램 틀의 높이를 낮춰 영상이 비추는 높이를 가운데로 되게끔 수정하는 방법밖에 없었다. 그렇기 때문에 내일 이 작업을 진행할 것이다.</p>		

개발일지			
날짜	11/14/2018	작성시간	17:59:00 PM
이름	조덕진		
개발과제	1. 홀로그램 출력 시험 및 홀로그램 수정작업		
내용	<p>1. 홀로그램 출력 시험 및 홀로그램 수정작업</p> <p>- 어제 1차적으로 홀로그램 제작을 완료하였고 오류/오차를 발견하였다. 그렇기 때문에 오늘 제작된 이 틀의 높이를 조정하고 이에 맞게 홀로그램 4분면 아크릴 틀 크기 조정을 하였다. 전체 높이를 반정도로 자르고 아크릴틀 또한 여기에 맞춰야 하므로 높이에 맞춰 피타고라스정의를 적용하여 다시 측정하여 조정하고 잘라 틀에 맞게 조정하였다.</p> <p>그 후 다시 모니터를 이용하여 Test를 하였다. 이번에는 전체적으로 크기가 줄었기 때문에 출력하는 화면영상에 맞게 4분면 아크릴 틀의 중앙에 맞게 비춰졌다. 전체적으로 화면을 출력하는 것이고 출력하는 화면 역시 시뮬레이션 프로그램이다보니 배경의 색이 밝았다. 이로인하여 홀로그램처럼 보이지는 않았지만 이것은 추후에 수정할 계획이다.</p> <p>2. 추가 동적 시뮬레이션(화재시 건물대피) 추가 및 구현을 위한 구상/환경 제작</p> <p>- 홀로그램 틀 제작이 완료되고 나서 3차 우선순위작업에 전반적으로 완료된 상황이었다. 여유적인 시간이 남았기 때문에 추가적으로 기능을 구현하기로 하였다. 남은 시간 안에 최대한 구현을 완료하는 방향으로 작업을 진행하기로 하였으며 완료되면 적용을 시키고 불가피하게 구현이 완료되지 못하게 되면 현재까지 구현이 완료된 기능들만 프로젝트에 포함시키기로 하였다.</p> <p>나는 추가적으로 동적상황요소(건물화재)에서 특별한 상황을 구현하기로 하였다. 건물화재 상황일 경우 화재가 난 건물의 내부로 들어가 사람들이 대피하는 인공지능 알고리즘을 구현할 예정이다. 그렇기 때문에 알고리즘 구현을 위한 디자인 및 구상을 하는 동시에 건물 내부에 대한 환경을 제작을 하였으며 제작완료하여 알고리즘 구현이 남았다.</p>		

개발일지			
날짜	11/15/2018	작성시간	20:00:00 PM
이름	조덕진		
개발과제	면접으로 인한 프로젝트 불참		
내용	<p>면접으로 인한 프로젝트 불참</p> <p>- 개인적으로 취업활동을 동시에 진행하고 있으며 면접일정으로 인해 프로젝트에 불참하였다.</p>		

개발일지			
날짜	11/16/2018	작성시간	18:15:00 PM
이름	조덕진		
개발과제	1. 현재까지 구현된 Project program 오류검출 및 최적화		
내용	<p>1. 현재까지 구현된 Project program 오류검출 및 최적화</p> <p>- 오늘 반 전체적으로 기업에 면접을 보러가는 일정이 있었다. 나는 그 기업이 내가 희망하는 진로 또는 업무와 달랐기 때문에 남아서 프로젝트를 진행하였다.</p> <p>월요일에 3차 발표가 있었기 때문에 다른 작업을 하기보다는 현재까지 구현 완료된 프로젝트를 Test하며 오류를 검출하였다. 어제 사람 및 차량 AI의 자연스러운 움직임을 위한 퍼지로지직 기능 구현이 다른 팀원으로 부터 완료되었기 때문에 이를 현재까지의 프로젝트에 병합시켜 구동해보고 최적화 하였다. 추가적을 스크립트를 추가하고 기존의 스크립트도 수정하였으며 해당 사람AI 또는 차량AI에 수치를 조정하여 병합작업을 하였으며 완료하였다. 이외에도 또 다른 팀원에 의해 수정된 LeapMotion 이벤트 처리 기능을 병합하였다. 기존의 문제로는 Build시 LeapMotion의 좌표가 원인을 모르게 틀어지는 현상이 있었다. 이 오류는 특정 Build 할 시에만 발생하는 오류로써 원인을 찾지 못하였었다. 그렇기 때문에 구동되는 것을 목적으로 이를 Collider을 이용하여 터치하는 방식으로 수정하였고 오늘 나는 이러한 수정된 작업을 기존의 프로젝트에 병합하고 대체시켜 적용하고 최적화하였다.</p> <p>결과적으로 3차발표를 위한 전체적인 최적화 작업을 하였다. 또한 우리팀은 초기 프로젝트 목표로 계획했던 기능을 모두 구현하고 완료되었다. 프로젝트 기간이 2주정도 여유가 생겼기 때문에 토의를 통해 결정했던 추가적인 동적상황요소를 구현할 계획이다.</p>		

개발일지			
날짜	11/19/2018	작성시간	18:15:00 PM
이름	조덕진		
개발과제	1. 3차 발표를 위한 발표준비 및 ppt제작/3차발표		
내용	<p>1. 3차 발표를 위한 발표준비 및 ppt제작/3차발표</p> <p>- 오늘은 3차 발표를 하는 날이다. 그렇기 때문에 발표를 위한 빌드 작업 및 프로그램 최적화 작업을 하였다. 우선적으로는 2차에 비하여 추가된 사항에 대하여 기존 발표 PPT에 추가하여 발표 PPT를 완성시켰다. 추가된 사항으로는 초기에 개발하였던 LeapMotion 기능을 추가시켜 특정 상황(시뮬레이션ON모드) 일때 사용할 수 있도록 추가하였다. 그리고 상황변화요소에 "유성낙하" 상황을 구현한 것을 추가시켰다. 또한 사람 AI와 차량AI의 자연스러운 움직임을 위하여 구현한 퍼지로지 기능을 기존의 프로젝트에 추가시켰다.</p> <p>제작한 홀로그램 틀을 이용하여 구현된 모든 기능을 병합시킨 프로젝트를 구동을 시켜 시뮬레이션 하는 동시에 출력되게 하였다.</p> <p>발표를 하던 도중 시연을 하는 과정에서 TrinusVR을 이용하여 VR기기와 연동이 되지 않았었다. 개발단계에서는 아무런 문제가 없이 잘 연동되었기 때문에 일시적인 기기 오류 또는 PC와 휴대폰의 연결 문제로 판단하였다. 그렇기때문에 이후 Test를 계속적으로 진행하며 점검할 예정이다.</p>		

개발일지			
날짜	11/20/2018	작성시간	18:45:00 PM
이름	조덕진		
개발과제	1. 추가 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현시작		
내용	<p>1. 추가 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현</p> <p>- 3차 발표를 마치고 2주간의 여유시간이 있기 때문에 기능을 추가하기로 하였다. 나는 상황변화요소중 건물화재에서 추가적인 요소, 화재시 건물내부 대피 상황을 구현하기로 하여 이를 구상하고 제작하는 것을 시작하였다. 상황변화요소 중 건물화재에서 한단계 더 선택할 수 있는 사항을 둘 예정이다. 우선적으로는 시뮬레이션 구동 중인 기존의 외부 카메라 시점과 건물 내부를 살펴볼수 있는 내부 카메라 시점으로 구분지은 다음 외부 카메라시점은 구현이 이미 되어있으므로 건물 내부를 제작을 하기 시작하였다.</p> <p>이를 위해 건물 내부에 해당하는 환경을 제작을 하였다. Asset들을 이용하여 3층짜리의 건물 내부 환경을 제작을 완료하였고 대피 알고리즘 구상 및 구현을 시작하였다. 건물 내부를 살펴보는 모드를 선택 했을 시에 UI 버튼을 두 개를 생성해 한개는 일반상황 또다른 하나는 화재 상황으로 이 버튼을 눌렀을 때 사람 AI의 행동변화를 주는 것으로 제작을 하였다. 일반 상황 시에는 자유롭게 건물 내부를 움직이다가 화재시에는 대피로를 통해 빠르게 건물 밖으로 대피하는 구조로 알고리즘을 작성하고 스크립트를 만들었다.</p> <p>추가적으로 사람AI 끼리 만났을 때, 대피하는 상황에서 병목현상이 발생했을 때, 연령대 별로 대피하는 순서 등 구체적인 조건사항을 추가 시킬 예정이다.</p>		

개발일지			
날짜	11/21/2018	작성시간	17:50:00 PM
이름	조덕진		
개발과제	<ol style="list-style-type: none"> 1. 완료보고서를 위한 회의 및 작성 2. AI움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 3. 수정된 퍼지로지 AI 움직임을 기존 Project에 병합 		
내용	<ol style="list-style-type: none"> 1. 완료보고서를 위한 회의 및 작성 <ul style="list-style-type: none"> - 오늘부터 완료보고서 작성에 들어갔다. 전체적인 구성을 위하여 회의를 하였다. 전체적인 구조를 구성하였고 이를 위하여 자료를 추가 및 수정작업을 하였다. 프로그램 개요, 개발일지, 구현 기술 등 현재까지 사용하고 구현한 것을 바탕으로 자료를 추가하고 있다. 2. AI움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 <ul style="list-style-type: none"> - 어제에 이어서 추가 동적 상황 변화 요소를 구현하였다. 건물내부 대피 중 일반상황일 때는 사람AI가 네비게이션 기능이 아닌 건물 내부의 특정 지점을 향하여 자유롭게 움직이도록 하였으며 화재 발생시에는 대피로를 따라 건물 밖으로 대피하도록 구현하였다. 특히 화재 발생시에는 현재의 위치로 부터 가장 가까운 대피 지점을 찾고 찾은 후에는 대피경로를 따라 이동하게 하였다. 사람AI의 움직임을 더욱 자연스럽게 만들 예정이고 움직임, 이동 속도에 따른 애니메이션 구현도 할 계획이며, 이동 및 대피 시에 발생하는 상황조건들에 대하여도 알고리즘을 추가할 예정이다. 3. 수정된 퍼지로지 AI 움직임을 기존 Project에 병합 <ul style="list-style-type: none"> - 팀원중 한명은 3차 발표이후 차량 AI 움직임에 대하여 더욱 자연스럽게 현실반영을 위해 이를 보수 수정하는 작업을 하였으며 최대한 구현하여 작업을 완료하였다. 이 완료한 작업을 현재까지 기능 구현되고 병합되어진 통합 프로젝트 프로그램에 접목시켰다. 수정된 스크립트에 맞게 차량AI의 3D Object들의 구성요소를 수정하였으며 새로운 스크립트를 추가하여 적용시켰다. 		

개발일지			
날짜	11/22/2018	작성시간	18:25:00 PM
이름	조덕진		
개발과제	1. 완료보고서를 위한 회의 및 작성 2. AI움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현		
내용	1. 완료보고서를 위한 회의 및 작성 - 완료보고서 작성 작업을 계속 진행 하였다. 프로젝트 시작 전에 전체 예상 기획서를 참고로 하여 작성을 이어나갔으며 예상 기획서와 다르게 실제 진행하면서 변경된 작업 및 추가/삭제된 기능들을 수정하여 제작사실 바탕으로 작성하였다. 또한 개발일지도 정리하고 아키텍처를 수정하여 추가할예정이다. 2. AI움직임 동적 시뮬레이션(화재시 건물내부 대피) 추가 및 구현 - 건물 내부의 사람AI의 움직임을 자연스럽게 수정하였다. 우선적으로는 사람AI들이 자유롭게 자동적으로 이동을 하도록 하였으며 사람AI끼리 부딪혔을 경우에는 서로의 우선순위를 비교하여 이동을 계속하거나 제자리에 잠시동안 멈추도록 구현을 하였다. 이는 대피시에도 똑같이 적용이 되고 연령대별로 아이, 노인, 청년 기준으로 일반 상황과 대피상황 모두 우선적으로 지나가도록 하는 목적으로 구현한 것이다. 대피상황 시에 대피로에 생기는 병목현상을 해결하고자 구현한 목적도 있다. 또한 대피상황과 일반상황에 움직이는 속도를 달리 설정하였으며 속도에 따라 애니메이션 모션도 다르게 적용하여 조금더 자연스럽게 움직이도록 구현하였다.		

개발일지			
날짜	11/23/2018	작성시간	18:05:00 PM
이름	조덕진		
개발과제	1. 완료보고서를 위한 회의 및 작성 2. 현재 프로젝트 Test 실행 후 오류 검출 및 최적화		
내용	1. 완료보고서를 위한 회의 및 작성 - 오늘까지의 개인개발일지, 개별개발일지가 전체 완료보고서에 추가되므로 작성을 완료하고 현재까지 작성된 일지들을 전부 완료보고서 양식에 맞게 추가하였다. 그리고 프로젝트에 구현된 기능들을 기능별로 분류하고 작성된 중요 스크립트 역시 완료보고서에 추가하였다. 팀원 각 각은 프로젝트를 통해 느낀점 및 소감을 주말에 작성을 하고 월요일날 최종적으로 완료보고서 작성을 마무리할 계획이다. 2. 현재 프로젝트 Test 실행 후 오류 검출 및 최적화 - 프로젝트를 시작한 시점부터 회의를 통해 결정하였던 기능 구현이 오늘 기준으로 완료가 되었다. 프로젝트 진행도중에 추가된 기능들도 있고 처음부터 목표로 구현하기로 하였던 기능들도 있었다. 우리 팀은 전체 프로젝트 일정에 맞게 기능 구현을 할 수 있었고 다음주부터는 이를 다양한 경우의 수, 다양한 방법으로 마지막 발표까지 지속적으로 Test하며 오류를 검출을 할 예정이며 검출된 오류를 수정하고 최대한 최적화할 계획이다.		

iii. 이태경 개발일지

개발일지			
날짜	2018.09.27	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도 설계 세부사항 1. Canvas_Inventory 설계 및 디자인구성 2. Canvas_Selection 설계 및 디자인구성		
내용	1. Canvas_Inventory 구조도 설계 2. Canvas_Inventory 카테고리 항목배치 3. Canvas_Inventory 디자인 시작		

개발일지			
날짜	2018.09.28	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. Canvas_Inventory 설계 및 디자인구성 2. Canvas_Selection 설계 및 디자인구성 3. Canvas_Item_Menu 설계 및 디자인 구성		
내용	1. Canvas_Inventory 틀제작 2. Canvas_Inventory 카테고리별 기능 코드 구현 3. Canvas_Inventory Object_List 등록 및 구동 테스트		

개발일지			
날짜	2018.10.01	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. Canvas_Inventory & Canvas_Selection 연동 3. Canvas_Item_Menu 설계 및 디자인 구성 (수정 할것)		
내용	1. Canvas_Inventory 및 Canvas_Select 설계 완료 및 두 UI 결합 2. Object 선택 시 띄워지는 Canvas_Item_Menu 디자인 및 설계		

개발일지			
날짜	2018.10.02	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. Canvas_Item_Menu 설계 및 디자인 2. UI 테스트		
내용	1. Canvas_Inventory 및 Canvas_Select 설계 완료 및 두 UI 결합 완료 2. Object 선택 시 띄워지는 Canvas_Item_Menu 디자인 완료 및 Object 개체 선택시 해당 정보 출력		

개발일지			
날짜	2018.10.04	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. 전체적인 UI 조정 2. UI 테스트 3. AI 동적 패턴 알고리즘 구상		
내용	1. Canvas_Item_Menu 구성 조정 2. AI 요소 및 동적 움직임 기능 알고리즘 구상 3. 네비게이션 효과 기능 구상		

개발일지			
날짜	2018.10.05	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. UI 테스트 2. AI 동적 패턴 알고리즘 구상 3. 길찾기 알고리즘(A*)와 Unity Navigation 조사		
내용	1. UI 구성 및 조정 최종 테스트 2. AI 요소 및 동적 움직임 기능 알고리즘 구상 3. 길찾기 알고리즘(A*)와 Unity Navigation 조사		

개발일지			
날짜	2018.10.08	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	1. AI 요소 및 동적 움직임 기능 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 상태 머신조사		

개발일지			
날짜	2018.10.10	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	1. Navigation Bake를 Source code로 구현 방식 검색 2. UnityEngine.AI 안의 NaviMesh 함수 조사 3. AI_Test Scene 제작 및 관련 UI 제작		

개발일지			
날짜	2018.10.11	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	길찾기 알고리즘(A*)와 Unity Navigation 조사 - NavMeshLink, NavMeshModifier, NavMeshModifierVolume, NavMeshSurface C# 코드 분석 AI 동적 패턴 알고리즘 구상 - LTK_AI_Road_Baking.cs 작성 및 테스트 Scene 준비 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사		

개발일지			
날짜	2018.10.12	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	AI 동적 패턴 알고리즘 구상 - 자동 Navigation bake 완료 - 다수의 캐릭터 생성 움직임 구현 - 캐릭터의 상태, 이상 효과 구비 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사 동적 구동 UI 생성 및 디자인		

개발일지			
날짜	2018.10.15	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도 설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	AI 동적 패턴 알고리즘 구상 - 자동 Navigation bake 구현 완료 - 다수의 캐릭터 생성 애니메이션 준비 - 캐릭터의 상태, 이상 효과 구비 - 캐릭터와 차량의 AI 구분 및 차도와 인도의 구분을 할 것이 요소를 검색 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사		

개발일지			
날짜	2018.10.16	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 길찾기 알고리즘(A*)와 Unity Navigation 조사 3. 캐릭터 행동 패턴 및 군집 효과 자료 조사 1차 발표 준비		
내용	AI 동적 패턴 알고리즘 구상 - 다수의 캐릭터 생성 Run, Walk, Stop 등 각각의 상태 및 그에 따른 애니메이션 효과 적용 - 캐릭터와 차량의 AI 구분 및 차도와 인도의 구분을 할 것이 요소를 검색 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사 1차 발표 준비 - PPT 조율 및 PPT 작성		

개발일지			
날짜	2018.10.17	작성시간	PM 6:00
이름	이태경		
개발과제	1차 발표		
내용	1차 발표 UI 구성 및 구조 시연 및 발표		

개발일지			
날짜	2018.10.18	작성시간	PM 6:00
이름	이태경		
개발과제	UI 구성 및 구조도설계 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	AI 동적 패턴 알고리즘 구상 - 다수의 캐릭터 생성 Run, Walk, Stop 등 각각의 상태 및 그에 따른 애니메이션 효과 적용 - LTK_Road_***_Bake 코드를 통한 캐릭터와 차량의 AI 구분 및 차도와 인도의 구분 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사		

개발일지			
날짜	2018.10.19	작성시간	PM 6:00
이름	이태경		
개발과제	1. UI 구성 및 구조도 설계 2. AI 알고리즘 및 패턴 구상 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	AI 동적 패턴 알고리즘 구상 - 다수의 캐릭터 생성 Run, Walk, Stop 등 각각의 상태 및 그에 따른 애니메이션 효과 적용 - LTK_Road_***_Bake 코드를 통한 캐릭터와 차량의 AI 구분 및 차도와 인도의 구분 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사		

개발일지			
날짜	2018.10.22	작성시간	PM 6:00
이름	이태경		
개발과제	1. UI 구성 및 구조도 설계 2. AI 알고리즘 및 패턴 구상 세부사항 1. AI 동적 패턴 알고리즘 구상 2. 캐릭터 행동 패턴 및 군집 효과 자료 조사		
내용	AI 동적 패턴 알고리즘 구상 - 완성된 Character AI와 Car AI를 메인 프로젝트에 결합 캐릭터 행동 패턴 및 군집 효과 자료 조사 - 유니티 게임 AI 프로그래밍 2/e 통한 군집 효과 및 캐릭터 상태 행동 패턴 조사		

개발일지			
날짜	2018.10.23	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정(캐릭터)		
내용	1. AI_테스트 및 Animation 효과 조정 - LTK_AI_Human_Move_Point.cs 추가 및 LTK_AI_Human_Move.cs의 기능 분할 - Enum구문을 통한 상황에 따른 움직임을 조정		

개발일지			
날짜	2018.10.24	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정(차량)		
내용	1. AI_테스트 및 Animation 효과 조정 - LTK_AI_Car_Move_Test 차량의 움직임 구현 및 테스트 - LTK_AI_Human_Move_Point.cs 추가 및 LTK_AI_Human_Move.cs의 기능 분할 - Enum구문을 통한 상황에 따른 움직임을 조정		

개발일지			
날짜	2018.10.25	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정(차량)		
내용	1. AI_테스트 및 Animation 효과 조정 - LTK_AI_Car_Move_Test.cs의 분할 이동 코드의 LTK_AI_Car_Move.cs와 도착 지점의 LTK_AI_Car_Move_Point.cs로 각 기능을 구현 - LTK_AI_Human_Move_Point.cs 추가 및 LTK_AI_Human_Move.cs의 기능 분할 - Enum구문을 통한 상황에 따른 움직임을 조정		

개발일지			
날짜	2018.10.26	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정		
내용	1. AI_테스트 및 Animation 효과 조정 - 3차 준비 및 퍼지 로직을 이용한 인공지능 AI 구상 및 준비 - 구현된 Car_AI 및 Human_AI 요소 테스트 및 버그 탐색		

개발일지			
날짜	2018.10.29	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정		
내용	1. AI_테스트 및 Animation 효과 조정 - 3차 준비 및 퍼지 로직을 이용한 인공지능 AI 구상 및 준비 - 구현된 Car_AI 및 Human_AI 요소 테스트 및 버그 탐색 - 만든 AI 코드와의 프로젝트와의 결합		

개발일지			
날짜	2018.10.30	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 2. 퍼지로지식 구상		
내용	1. AI_테스트 및 Animation 효과 조정 - 구현된 Car_AI 및 Human_AI 요소 테스트 및 버그 탐색 - 만든 AI 코드와의 프로젝트와의 결합 2. 퍼지로지식 - Human_AI의 상태 Walk와 Run의 분간되는 IsRun Bool 값을 조정할 퍼지로지식 준비 - Car_AI의 속도를 퍼지로지식을 통한 제어 준비		

개발일지			
날짜	2018.10.31	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 2. 퍼지로지식 구상		
내용	1. AI_테스트 및 Animation 효과 조정 - 구현된 Car_AI 및 Human_AI 요소 테스트 및 버그 탐색 - 결합된 프로젝트 내의 AI_Navigation 오류제어 2. 퍼지로지식 - Human_AI의 상태 Walk와 Run의 분간되는 IsRun Bool 값을 조정할 퍼지로지식 준비 - Car_AI의 속도를 퍼지로지식을 통한 제어 준비		

개발일지			
날짜	2018.11.01	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 2. 퍼지로지식 구상 3. Save/Load UI 디자인		
내용	1. AI_테스트 및 Animation 효과 조정 - 구현된 Car_AI 및 Human_AI 요소 테스트 및 버그 탐색 - 결합된 프로젝트 내의 AI_Navigation 오류제어 2. 퍼지로지식 - Human_AI의 상태 Walk와 Run의 분간되는 IsRun Bool 값을 조정할 퍼지로지식 준비 - Car_AI의 속도를 퍼지로지식을 통한 제어 준비 3. Save/Load UI 디자인		

개발일지			
날짜	2018.11.02	작성시간	PM 6:00
이름	이태경		
개발과제	1. 2차발표를 위한 Project 최적화 및 발표준비(ppt제작)		
내용	1. 2차발표를 위한 Project 최적화 및 발표준비(ppt제작) - 2차 작업 PPT 정리 및 AI_Navigation 테스트 화면 준비		

개발일지			
날짜	2018.11.05	작성시간	PM 6:00
이름	이태경		
개발과제	1. 2차발표		
내용	1. 2차발표 - 동적 요소인 AI 소개 및 설명		

개발일지			
날짜	2018.11.06	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 2. 퍼지로지직		
내용	1. AI_테스트 및 Animation 효과 조정 - AI_Animation 수정 및 LTK_AI_Human_Move와 LTK_AI_Human_State_Animation.cs 간의 오류 수정 - AI_Animator의 버그 수정 및 오류 제어 2. 퍼지로지직 - Human_AI의 상태 Walk와 Run의 분간되는 IsRun Bool 값을 조정할 퍼지로지직 준비 - Car_AI의 속도를 퍼지로지직을 통한 제어 준비		

개발일지			
날짜	2018.11.07	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정 2. 퍼지로지		
내용	1. AI_테스트 및 Animation 효과 조정 - AI_Animation 수정 및 LTK_AI_Human_Move와 LTK_AI_Human_State_Animation.cs 간의 오류 수정 - AI_Animatior의 버그 수정 및 오류 제어 2. 퍼지로지 - Human_AI_Fuzzy 작성 시작 및 캐릭터의 시작 거리와 끝점간의 거리를 계산하는 애니메이션 커브 등록		

개발일지			
날짜	2018.11.08	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정 2. 퍼지로지직		
내용	1. AI_테스트 및 Animation 효과 조정 - AI_Animation 수정 및 LTK_AI_Human_Move와 LTK_AI_Human_State_Animation.cs 간의 오류 수정 - AI_Animatior의 버그 수정 및 오류 제어 2. 퍼지로지직 - Human_AI_Fuzzy 작성 시작 및 캐릭터의 시작 거리와 끝점간의 거리를 계산하는 애니메이션 커브 등록 - 캐릭터 체력 및 체력 상태에 따른 애니메이션 커브 작성		

개발일지			
날짜	2018.11.09	작성시간	PM 6:00
이름	이태경		
개발과제	1. AI_테스트 및 Animation 효과 조정 2. 퍼지로지		
내용	1. AI_테스트 및 Animation 효과 조정 - AI_Animation 수정 및 LTK_AI_Human_Move와 LTK_AI_Human_State_Animation.cs 간의 오류 수정 - AI_Animatior의 버그 수정 및 오류 제어 2. 퍼지로지 - Human_AI_Fuzzy 작성 시작 및 캐릭터의 시작 거리와 끝점간의 거리를 계산하는 애니메이션 커브 등록 - 캐릭터 체력 및 체력 상태에 따른 애니메이션 커브 작성 - 두 커브의 값의 통계를 통한 퍼지화 구현 완료		

개발일지			
날짜	2018.11.12	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로직		
내용	1. 퍼지로직 - Car_AI_Fuzzy 준비 및구상 - 차량간의 퍼지화준비		

개발일지			
날짜	2018.11.13	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로직		
내용	1. 퍼지로직 - Car_AI_Fuzzy 준비 및구상 - 차량 안의 Layer 변수가 담긴 기지모 설치 - 차량의 90도 시야각인 Car_AI_Side_Mirror 함수 작성		

개발일지			
날짜	2018.11.14	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로지		
내용	1. 퍼지로지 - Car_AI_Fuzzy 준비 및 구상 - 차량 간격을 조정하는 애니메이션 커브 작성 - 차량의 방향과 거리를 받아 그에 따른 퍼지화 완료		

개발일지			
날짜	2018.11.15	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로지 2. 동영상 촬영		
내용	1. 퍼지로지 - 완료된 퍼지화 적용 및 테스트 - 테스트 완료된 Object와 프로젝트의 결합 시도 2. 동영상 촬영 - 3차 발표 준비 및 올릴 영상 촬영		

개발일지			
날짜	2018.11.16	작성시간	PM 6:00
이름	이태경		
개발과제	1. 3차 발표 준비		
내용	1. 3차 발표 준비		

개발일지			
날짜	2018.11.19	작성시간	PM 6:00
이름	이태경		
개발과제	1. 3차 발표		
내용	1. 3차 발표		

개발일지			
날짜	2018.11.20	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로직		
내용	1. 퍼지로직 - Car_AI_Fuzzy, Car_AI_Side_Mirror 소스코드 버그 탐색 및 수정 - 차량 Object의 Prafb의 조정 - 추가된 Layer인 Vehicle_Center를 Vehicle_Front, Vehicle_Back으로 세분화		

개발일지			
날짜	2018.11.21	작성시간	PM 6:00
이름	이태경		
개발과제	1. 퍼지로지 2. 완료보고서		
내용	1. 퍼지로지 - Car_AI_Fuzzy, Car_AI_Side_Mirror 소스코드 수정 완료 - 수정된 차량Object 테스트 - 수정된 차량 Object와 프로젝트 결합 2. 완료보고서 - Holo_City 프로젝트 완료보고서 작성		

개발일지			
날짜	2018.11.22	작성시간	PM 6:00
이름	이태경		
개발과제	1. 완료보고서		
내용	1. 완료보고서 - Holo_City 프로젝트 완료보고서 작성		

개발일지			
날짜	2018.11.23	작성시간	PM 6:00
이름	이태경		
개발과제	1. 완료보고서		
내용	1. 완료보고서 - Holo_City 프로젝트 완료보고서 작성		

4. 프로젝트 개발 내용

A. 프로젝트 배경 지식/기술/알고리즘

i. H/W [립모션 / 홀로그램]

1. 립모션



내 용	설 명
길 이	0.5 inches
너 비	1.2 inches
깊 이	3 inches
무 게	0.1 pounds
케이블 포함	60cm 또는 150cm USB 2.0
최소 시스템 사양	Window 7,8 또는 Mac OSX 10.7 Lion, AMD phenom2, Intel core i1, i5, i7 Processor USB2.0 port

- Michael Buckwald와 David Holz가 2010년 설립한 3D 모션컨트롤 기기 개발업체에서 만든 제스처 기반의 HID 기기
- 특징
 - 손가락 관절 하나하나와 손바닥의 움직임을 체크하여 영화에서 보던 것과 같이 화면을 제어 할 수 있는 제품
 - 두 손과 열 손가락을 0.01mm 정밀도, 200FPS (초당 200컷)의 속도로 추적이 가능
 - 손에 한해서는 손가락 마디 관절의 구부림까지 거의 완벽하게 인식



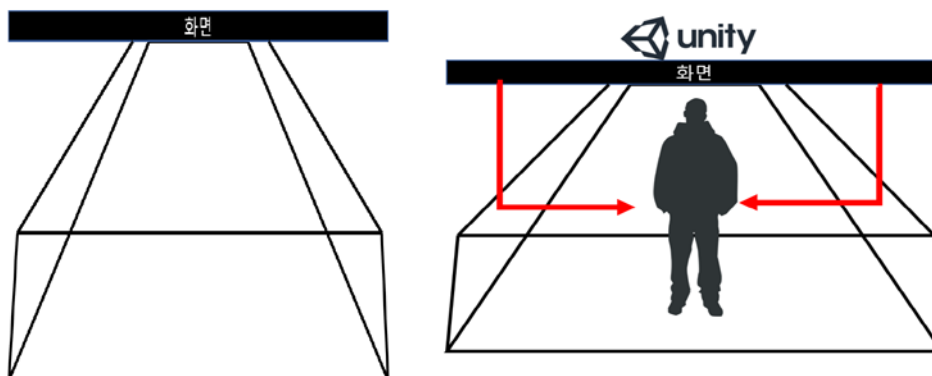
➤ LeapMotion의 동작 및 구성

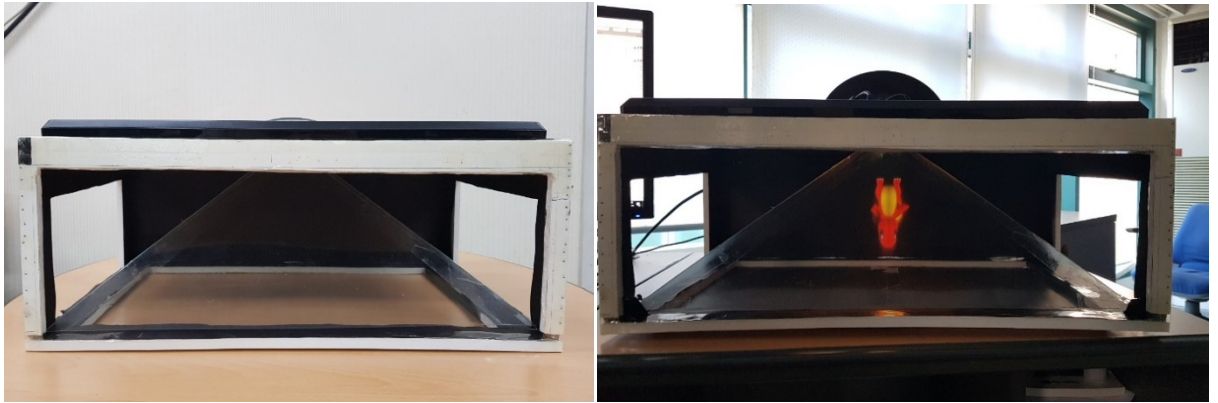
- 적외선 LED: 적외선 카메라를 위한 광원
- 적외선 카메라: 2대의 CMOS 적외선 인식 모듈을 적제한 깊이 인식
- 영상처리: IR 카메라로 촬영된 적외선 이미지를 토대로 손을 인식, 제스처에 맞게 움직이는 3D 핸드 모델을 구현

2. 홀로그램



- '완전한'이라는 의미의 'Holos'와 '정보, 메시지'라는 의미의 'Gramma'의 합성어
- 홀로그래피 원리를 이용하여 3차원으로 만들어진 입체적 시각 정보. 2개 이상의 레이저 광선의 간섭효과를 이용 간섭무늬가 저장된 필름
- ※ 홀로그래피: 레이저에서 나온 광선을 2개로 나눠 하나의 빛은 직접 스크린을 비추게 하고, 다른 하나의 빛은 관찰자가 보려고 하는 물체에 비추는 것
- 홀로그램의 구성 및 동작





- 화면: 홀로그램을 표현하는 영상 매체
- 홀로그램 패널: 화면에서 출력되는 영상의 빛의 굴곡을 생성 입체적인 홀로그램 표현
- 홀로그램 영상: Host / VR 시뮬레이션 화면을 홀로그램으로 실시간 출력
- Host / VR에 제공되는 시뮬레이션 Scene을 4분면으로 촬영 4분면으로 출력된 평면 화면을 홀로그램 패널에 비추어 나타나는 굴곡 현상을 이용 홀로그램을 통한 관찰자의 입체적인 시야를 제공

ii. S/W [비주얼 스튜디오 / Unity]

1. 비주얼 스튜디오

- 마이크로 소프트에서 개발한 통합 개발 환경
- 특징
 - 하나의 패키지 형태로 별도의 개발툴이 필요 없다.
 - Visual Basic(Visual Basic .NET), C++(+ Win32 API, MFC), C#(ASP.NET, WPF) 등등을 지원 그에 따른 UI제작 도구 제공
 - 최적화가 되어 있어 저사양 모드에서도 문제 없이 돌아 간다.

2. Unity

- 3D 및 2D 비디오 게임의 개발 환경을 제공하는 게임 엔진이자, 3D 애니메이션과 건축 시각화, 가상현실(VR) 등 인터랙티브 콘텐츠 제작을 위한 통합 저작 도구
- 특징
 - 27개의 플랫폼에서 사용 가능한 콘텐츠를 만들 수 있고, 제작 도구인 유니티 에디터는 윈도우와 맥OS를 지원
 - 게임 개발에 사용하는 스크립트 언어는 C#와 자바스크립트를 지원
 - 스크립트 작성은 유니티와 함께 설치되는 비주얼 스튜디오(맥OS의 경우, 모노디벨로프)를 이용하며, 다른 편집기와 연동하는 것도 가능

➤ 장점

- GUI가 직관적이며 툴 사용이 서툰 사람이 편하게 사용하기 좋다.
- 다양한 플랫폼으로 빌드가 가능하다.[단, 플랫폼의 특성에 맞게 어느정도의 조정 작업 및 최적화가 필요]
- 저렴한 라이선스 비용
- 애셋스토어의 존재로 인디게임 개발자 및 초보 개발자가 보다 쉽게 접근이 가능하다.

➤ 단점

- 부실한 한국어 지원
- 최적화가 부실하여 생기는 프레임 드랍
- 신 버전 릴리즈시 발생하는 버그 5.3 버전에 나타난 버그들이 많다.

iii. Language [C#]

- 마이크로소프트에서 개발한 객체지향 프로그래밍 언어, 닷넷 프레임 워크를 기반으로 만들어진 언어
- C# 특징
 - 닷넷 플랫폼을 가장 직접적으로 반영 하며 닷넷 플랫폼에 강하게 의존한다.
 - 문법적인 특성이 자바와 상당히 유사하며 닷넷 플랫폼 기술 역시 JVM과의 유사점이 많다.
- 본 프로젝트에서는 Unity 툴을 사용하는 대표적인 언어로 사용된다.

iv. Json

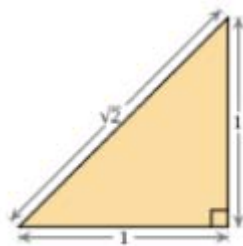
- 경량의 DATA-교환 형식이다. 사람이 읽고 쓰기에 용이하며, 기계가 분석하고 생성함에도 용이하다.
- 구조
 - name/value 형태의 쌍으로 collection 타입. 다양한 언어들에서, 이는 object, record, struct(구조체), dictionary, hash table, 키가 있는 list, 또는 연상배열로서 실현 되었다.
 - 값들의 순서화된 리스트. 대부분의 언어들에서, 이는 array, vector, list, 또는 sequence로서 실현 되었다.
- JSON 에서, 이러한 형식들을 가져간다
 - object는 name/value 쌍들의 비순서화된 SET이다.
 - object는 { (좌 중괄호)로 시작하고 } (우 중괄호)로 끝내어 표현한다. 각 name 뒤에 : (colon)을 붙이고 , (comma)로 name/value 쌍들 간을 구분한다.

v. **A* 알고리즘**

- 시작점과 목표점 사이의 최단 거리 길을 찾아주는 그래프/트리 탐색 알고리즘으로 게임에서 많이 사용되는 최단거리 길찾기 알고리즘
- 핵심 요소
 - 우선 순위 큐를 이용한 최소한의 비용의 경로를 선으로 탐색
 - 휴리스틱 추정값
 - Open List / Closed List를 이용한 노드 관리
- ※ 휴리스틱 추정값: 인간과 기계에서 어떤 문제를 해결하거나 제어하기 위해 필요한 정보를 위해 느슨하게 적용시키는 접근을 시도하는 전략

➤ 휴리스틱 추정값

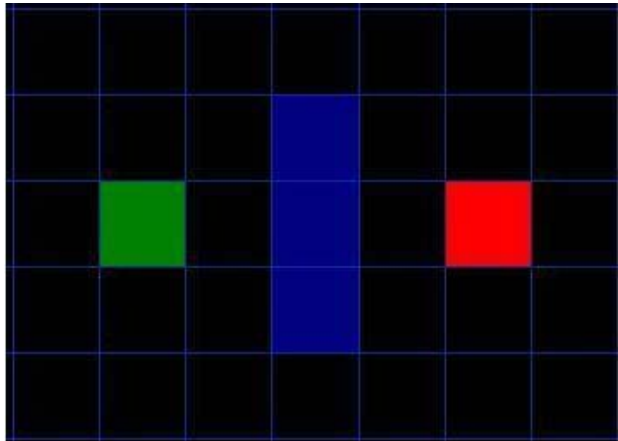
- $f(x) = h(x) + g(x)$



- $h(x)$: 출발노드 n 으로부터 도착 노드 n 까지의 경로 가중치[새로운 사각형까지의 거리]
- $g(x)$: 노드 n 으로 부터의 노드까지의 추정 경로 가중치[도착 노드까지의 예상 이동 비용]
- 맨허튼 거리 측정법 사용 $(x_1 - x_2) + (y_2 - y_1)$

➤ A* 알고리즘 동작 방식

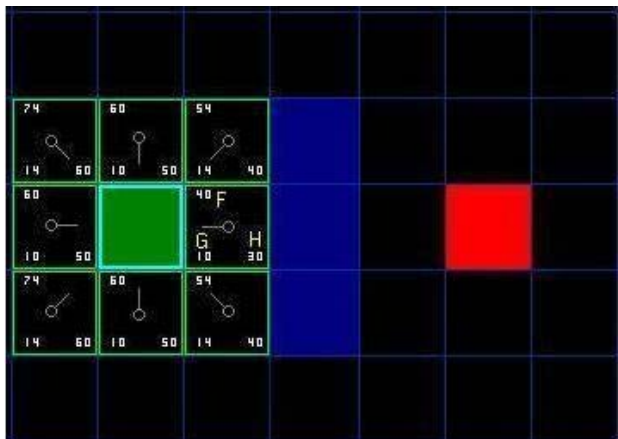
0. 설정



- 초록색: 출발노드
- 파란색: 장애물(이동 불가)
- 빨간색: 도착노드

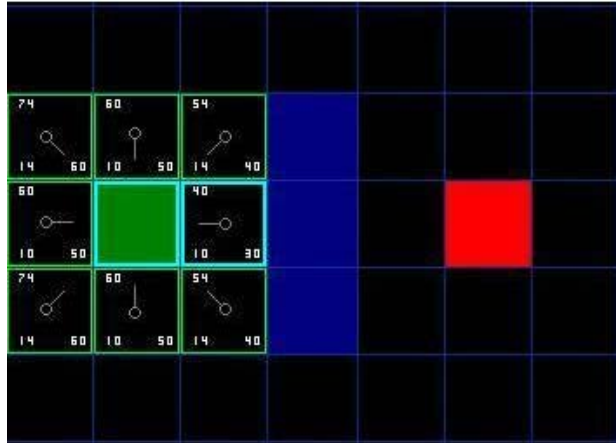
➔ 출발노드의 주변 노드 8개를 OpenList에 추가

1. 탐색시작 / 경로 채점



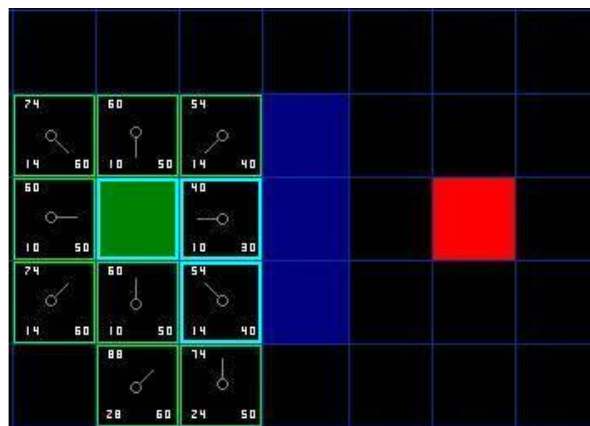
- ① Priority Queue에서 Pop()
- ② 출발 노드는 Closed List 에 추가
- ③ 출발 노드를 기준으로 주변 노드 8개를 탐색해 갈 수 있는 휴리스틱 추정값을 사용
가중치의 값과 부모노드[출발노드]를 구한다.
 - A. $G(x)$: [직선: 10, 대각선: 14]
 - B. $H(x)$: 장애물 까지 도착한 노드까지의 맨해튼 거리
- ④ 계산된 주변노드를 Open List에 추가

2. 경로 재 탐색

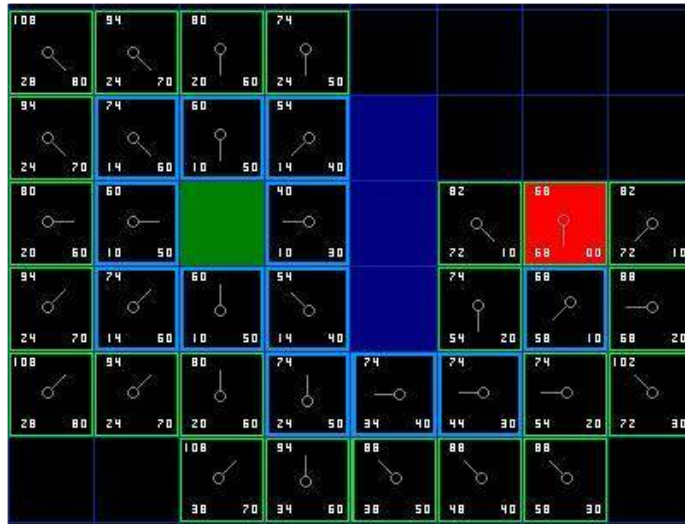


- ① $F(x)$ 값이 가장 낮은 40 노드를 Pop()
- ② 주변노드 검색
 - A. $F(x)=54$ 인 노드의 경로(\searrow)의 값과 $F(x)=40$ 을 통해 거쳐가는 ($\rightarrow \downarrow$) 경로의 값인 $G(x)$ 를 비교
 - B. $\rightarrow \downarrow$ 의 가중치가 $20[\rightarrow(G(x)=10)+\downarrow(G(x)=10)]$ \searrow 의 가중치가 14이므로 \searrow 의 경로를 탐색
- ③ 40노드는 Closed List에 Push()

3. 계속 탐색하기

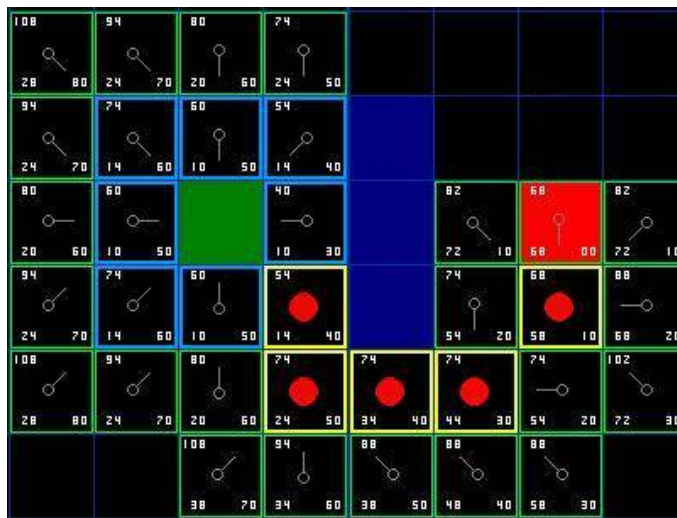


- ① 다시 Priority Queue에서 Pop()
- ② 54 노드 주위 노드 탐색
 - A. \leftarrow 노드는 Open List에 있기 때문에 $F(x)$ 값을 계산하지만 원래 가지고 있는 $F(x)$ 값이 더 작기 때문에 갱신되지 않는다.
- ③ 장애물에 걸리지 않으며 최소비용인 74 노드로 이동 하며 작업을 반복



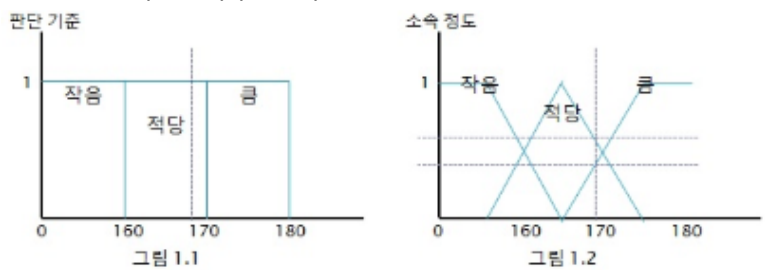
④ 붉은색의 도착 노드가 도달할 때까지 반복한다.

4. 결과출력

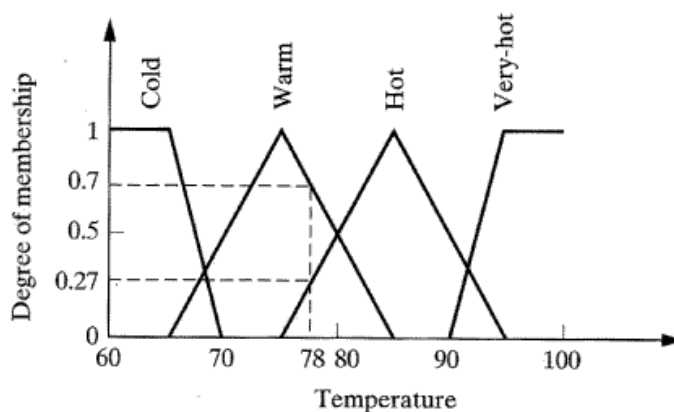


vi. Fuzzy Logic

- 퍼지논리는 어렵풋하고 애매모호한 인간의 언어[Ex]차갑다, 뜨겁다를 컴퓨터 언어로 표현하고자 한 이론
- 퍼지화
 - 제어시스템에서 측정된 정확한 입력값들을 퍼지 규칙에 의해 각각의 언어값과 소속 함수로 바꾸는 과정



- ※ 좌: 퍼지화 되지 않은 신장의 크기 정의
- ※ 우: 퍼지화 된 신장의 크기 정의
- 퍼지추론
 - 퍼지화를 통해 구한 소속값으로 결과를 추론하는 과정
 - 퍼지 입력을 퍼지규칙을 이용하여 퍼지 결과를 추론하는 과정
 - 퍼지추론 규칙기반: 시스템 제어기에 해당 진리표 로직에 기초하고 있다. 규칙기반은 퍼지집합과 입력 변수, 출력변수들과의 연관된 규칙의 집합이며 학 경우에 있어서 할 일을 결정해야 한다. 만약 두개 이상의 입력변수에 대해 출력변수를 결정하는 퍼지 규칙은 And 연산(두 값 중 최소값), OR 연산(두 값 중 최대값)이다.
 - 예시



- 기온이 78F 였을 때의 Warm 수치는 0.7 Hot 수치는 0.27의 소속값을 갖는다.

- 역 퍼지화

- 퍼지출력을 보통 수치로 변환 하는 과정
- 퍼지 출력값을 실제로 사용하기 위한 등가의 정확한 값으로 변환하는 것이다.
- 예시
 - 에어컨의 출력의 강도 설정을 낮게 중간 최고로 퍼지화를 가정
 - 규칙기반에서 평과한 결과값이 25% 이하이면 낮게 75% 이하이면 중간으로 결정하여 에어컨 제어 시스템에 전달, 이 단일 값을 역퍼지화라고 한다.

➤ 무게중심법

- 소속집합의 출력에 대한 증가 값을 구하기 위해 추력 소속 집합의 최대 단일 값과 각 출력 변수에 대한 소속 값을 곱하는 방식
- 무게중심법을 사용하여 분산화된 값의 평균을 잴 수 있다.
- 예시

$$\text{출력} = \frac{0.4 \times 30\% + 0.6 \times 50\%}{0.4 + 0.6} = 42\%$$

B. 프로젝트 상세 개발 내용

I. 기능정의 리스트

● Unity 기능 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
Unity	OS 설치	게임 엔진 설치	Ut-Oi-01	최신 버전 설치(2018.02.08)	공통	1
		Visual Studio 2017 설치	Ut-Oi-02	Unity 코딩을 위한 설치	공통	1
	VR 공간구현	Object 제작	Ut-Vm-01	시뮬레이션 내의 배치할 요소 제작 및 수집	윤혜진	1
		Terrain 제작	Ut-Vm-02	공간에 적절한 샘플 지형 제작	윤혜진	1
		물리 법칙 적용	Ut-Vm-03	물리 충돌 및 중력에 대한 Default 값 설정	김영서 조덕진	1
		조명 옵션 설정	Ut-Vm-04	시간 변화에 따른 조도 작성	김영서 조덕진	2
		효과음	Ut-Vm-05	UI 이벤트 효과음, 상황 부여에 따른 효과음	김영서	2
		AI(사람, 자동차) 패턴 AI 적용	Ut-Vm-06	상황에 따른 동적 요소(패턴 인식 AI) 설계	이태경 조덕진	2
	기능구현	Leap Motion 적용	Ut-Fo-01	Leap Motion 입력 데이터 전송 기능을 확인	김도현 김영서	2
		VR 빌드	Ut-Fo-02	프로젝트를 UR 콘텐츠 기반으로 빌드 기능을 확인	김도현 김영서	3
		메인 UI 구성	Ut-Fo-03	메인 UI 제작	이태경	1

● 디바이스 I/O 기능 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
디바이스, I/O Control	VR	Trinus APK 설치	Dc-Vr-01	Trinus APK 설치	김도현 김영서	3
		VR 시뮬레이션 화면 띄우기	Dc-Vr-02	Unity를 이용한 완성된 프로그램을 Trinus VR과 연동 확인	김도현 김영서	3
	홀로그램	Unity와 연동	Dc-Ho-01	Unity 건축 시뮬레이션 장면을 실시간 홀로그램 영상으로 송출	윤혜진 이태경	3
		홀로그램 장치 제작	Dc-Ho-02	홀로그램을 구현할 수 있는 장치 제작	윤혜진 이태경	3
	Leap Motion	모션 인식을 위한 함수 구현 / 연결	Dc-Lm-0 1	Leap을 이용해 손 위치 좌표 및 손 동작을 인식하여 사용자가 원하는 지시를 분간 지을 수 있게 한다.	김도현 김영서	2
		Unity와 연동	Dc-Lm-0 2	Unity와 연동 사용자의 손동작을 인식하여 Unity에서 표현할 수 있도록 연동	김도현 김영서	2
		Leap Motion 설치	Dc-Lm-0 3	Leap Motion 작동을 위한 SDK 설치	김도현 김영서	1

● Save & Load 기능 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
Save / Load	Json 파일 변환	Scene 업로드	Ws-Ab-01	시뮬레이션 Scene 정보를 저장	김도현 조덕진	2
		Scene 다운로드	Ws-Ab-02	저장된 Scene 정보를 Import	김도현 조덕진	2
		Object 업로드	Ws-Ab-03	사용자로부터 추가된 Object 정보를 저장	김도현 조덕진	2
		Object 다운로드	Ws-Ab-04	저장된 Object 정보를 Import	김도현 조덕진	2

● 우선 순위 리스트

가) 1순위 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
Unity	OS 설치	게임엔진 설치	Ut-Oi-01	최신 버전 설치(2018.02.08)	공통	1
		Visual Studio 2017 설치	Ut-Oi-02	Unity 코딩을 위한 설치	공통	1
	VR 공간구현	Object 제작	Ut-Vm-01	시뮬레이션 내의 배치할 요소 제작 및 수집	윤혜진	1
		Terrain 제작	Ut-Vm-02	공간에 적절한 샘플 지형 제작	윤혜진	1
		물리법칙 적용	Ut-Vm-03	물리 충돌 및 중력에 대한 Default 값 설정	김영서 조덕진	1
	기능구현	메인 UI 구성	Ut-Fo-03	메인 UI 제작	이태경	1
디바이스, I/O Control	Leap Motion	Leap Motion 설치	Dc-Lm-03	Leap Motion 작동을 위한 SDK 설치	김도현 김영서	1

나) 2순위 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
Unity	VR 공간구현	조명 옵션 설정	Ut-Vm-04	시간 변화에 따른 조도 작성	김영서 조덕진	2
		효과음	Ut-Vm-05	UI 이벤트 효과음, 상황 부여에 따른 효과음	김영서	2
		AI(사람, 자동차) 패턴 AI 적용	Ut-Vm-06	상황에 따른 동적 요소(패턴 인식 AI) 설계	이태경 조덕진	2
	기능구현	Leap Motion 적용	Ut-Fo-01	Leap Motion 입력 데이터 전송 기능을 확인	김도현 김영서	2
		AI movement 설계	Ut-Fo-04	특정 상황에 따른 동적 움직임 구현	이태경 조덕진	2
		Scene, Object Import	Ut-Fo-06	Asset Bundles 를 통한 Scene, Object Import	김도현 조덕진	2
디바이스, I/O Control	Leap Motion	모션 인식을 위한 함수 구현 / 연결	Dc-Lm-01	Leap을 이용해 손 위치 좌표 및 손 동작을 인식하여 사용자가 원하는 지시를 분간 지을 수 있게 한다.	김도현 김영서	2
		Unity와 연동	Dc-Lm-02	Unity와 연동 사용자의 손동작을 인식하여 Unity에서 표현할 수 있도록 연동	김도현 김영서	2
Save /	Json 파일 변환	Scene 업로드	Ws-Ab-01	시뮬레이션 Scene 정보를 저장	김도현 조덕진	2

Load	Scene 다운 로드	Ws-Ab-02	저장된 Scene 정보를 Import	김도현 조덕진	2
	Object 업 로드	Ws-Ab-03	사용자로부터 추가된 Object 정보를 저장	김도현 조덕진	2
	Object 다운 로드	Ws-Ab-04	저장된 Object 정보를 Import	김도현 조덕진	2

다) 3순위 리스트

대분류	중간항목	상세항목	항목번호	설명	담당자	순위
Unity	기능구현	VR 빌드	Ut-Fo-02	프로젝트를 UR 콘텐츠 기반으로 빌드 기능을 확인	김도현 김영서	3
		Effect 적용	Ut-Fo-05	VI 와 이벤트를 통한 Effect 적용	김영서	3
디바이스, I/O Control	VR	Trinus APK 설치	Dc-Vr-01	Trinus APK 설치	김도현 김영서	3
		VR 시뮬레이션 화면 띄우기	Dc-Vr-02	Unity를 이용한 완성된 프로그램을 Trinus VR과 연동 확인	김도현 김영서	3
	홀로그램	Unity와 연동	Dc-Ho-01	Unity 건축 시뮬레이션 장면을 실시간 홀로그램 영상으로 송출	윤혜진 이태경	3
		홀로그램 장치 제작	Dc-Ho-02	홀로그램을 구현할 수 있는 장치 제작	윤혜진 이태경	3

II. 주요 소스코드

➤ UI 모드[OFF모드]

i. InterFace 창

Inventory_Database 주요 기능

➤ LTK_Item_Info.cs의 변수

```

public string Item_name;           //아이템의 이름

public int Item_Id;               //아이템의 아이디

[TextArea]

public string Item_info;          //아이템의 정보

public Sprite Item_Image;         //아이템의 이미지

public GameObject Item_Model;     //Prefab화 시킨 Object

```

➤ LTK_Inventory_Item DataBase.cs의 주요기능

//Obj_data_temas에 Prefab화 시킨 Object를 테마에 맞추어 분할 저장

```

public GameObject[] obj_data_temas;

public List<Item> Background_ItemDB;    //배경   데이터베이스

public List<Item> Building_ItemDB;      //건물   데이터베이스

public List<Item> Road_ItemDB;          //길     데이터베이스

public List<Item> Structure_ItemDB;     //구조물 데이터베이스

public List<Item> Nature_ItemDB;        //자연   데이터베이스

public List<Item> Vehicle_ItemDB;       //차량   데이터베이스

public List<Item> Character_ItemDB;     //캐릭터 데이터베이스

```

```
public List<Item> Other_ItemDB;           //제작   데이터베이스

void SpliteList()
{
    (중략...)

    //Object에 담겨진를 테마를 검색

    switch (obj_tema.tag)
    {
        //만약 Prefab 테마가 배경일 경우

        case "Background":
            InitItemList(Background_ItemDB, obj_data);

            break;

        (중략...)
    }
}

// 해당하는 테마의 데이터 베이스에 아이템 구조에 맞게 넣는 작업을 한다.

void InitItemList(List<Item> ItemDB, LTK_Item_Info Item_info) {
    Item Inven_Item = new Item();

    Inven_Item.itemID = Item_info.Item_Id;

    Inven_Item.itemName = Item_info.Item_name;

    Inven_Item.itemDesc = Item_info.Item_info;

    Inven_Item.itemIcon = Item_info.Item_Image;
```

```
Inven_Item.itemModel = Item_info.Item_Model;  
  
ItemDB.Add(Inven_Item);  
  
}
```

Inventory 주요 기능

➤ LTK_Inventory.cs의 주요기능

```
LTK_Inventory_Item_DaBase DataBase;
```

//버튼으로 입력 받은 String을 검색

```
public void ViewList(string tag_name)
```

```
    switch (tag_name)
```

```
    {
```

//저장된 인벤토리 데이터 베이스 테마와 받아온 Tag_name이 일치할 경우

//해당하는 테마의 데이터 베이스를 넘겨준다.

```
        case "Background":
```

```
            DataList(DataBase.Background_ItemDB);
```

```
            Inven_UI.text = "Background";
```

```
            break;
```

(중략...)

```
    }
```

```
void DataList(List<Item> Inven_Items)
```

```
{
```

```
    Save_List = Inven_Items;
```

//페이지 출력

```
DataListPage(Inven_Items.Count);

ResetViewData();

//슬롯에게 보내 주는 아이템 정보

ViewData(INIT_Page);
}

void ViewData(int Page)
{
    int ViewListPage = Page - 1;

    int count = 1 * ViewListPage * SlotMAX;

    //각 슬롯마다 순서대로 아이템의 정보를 넘겨준다.

    foreach (LTK_Slot_Menu SlotData in Datas)
    {
        if (Save_List.Count <= count)
        {
            break;
        }

        SlotData.icon_Setting(Save_List[count]);

        count++;
    }
}
```

```
➤ LTK_Slot.cs
// 인벤토리에서 넘긴 아이템 정보를 저장

Item inven_item;
```

(중략...)

```
public void icon_Setting(Item item)
{
    inven_item = item;

    //아이템 이미지를 슬롯 이미지로 출력한다

    Slot.ShowImage(item.itemIcon);
}
```

On/OFF 모드 중요 기능

➤ LTK_OnOff.cs 기능

//UI와 AI모드에 사용되는 캔버스를 배열로 분할 저장

```
public GameObject[] canvas_Kind;
```

(중략...)

```
public void OnClick_OnOff(GameObject input_Field)
{
    if (input_Field.tag == "On")
    {
        if (!LTK_Save.OnOff)
        {
            //On/Off버튼 색상 변경

            input_On(input_Field);

            //AI 모드 캔버스 활성화

            Canvas_AI_Enable();
        }
    }
}
```

```
        //Debug.Log("On입력");

    }

}

else if(input_Field.tag == "Off")
{
    if (LTK_Save.OnOff)
    {

        AI_Menu_Button_Reset();

        //On/Off버튼 색상 변경
        input_Off(input_Field);

        //UI 모드 캔버스 활성화
        Canvas_UI_Enable();

        //Debug.Log("Off입력");
    }
}

}
```

- AI 이동
 - 캐릭터의 이동

A* 알고리즘

- 캐릭터가 이동 가능한 경로 생성

- LTK_Human_Road_Bake.cs의 주요기능

```
void UpdateNavMesh(bool asyncUpdate = false)
```

```
{
```

```
    //Tag가 선언된 곳 길 Object의 모음
```

```
    LTK_AI_Human_Road_Tag.Collect(ref m_Sources);
```

```
    //Agent Type[캐릭터(1) OR 차량(2)]의 인덱스 번호를 읽어 해당 Type을 읽어 변수에 저장한다.
```

```
    var AppendBuildSettings = NavMesh.GetSettingsByIndex(1);
```

```
    var bounds = QuantizedBounds();
```

```
    if (asyncUpdate)
```

```
        m_Operation = NavMeshBuilder.UpdateNavMeshDataAsync(m_NavMesh, AppendBuildSettings, m_Sources, bounds);
```

```
    else
```

```
        //저장된 변수와 모아진 길 Object를 Bake를 해서 경로로 인식하게 만든다.
```

```
        NavMeshBuilder.UpdateNavMeshData(m_NavMesh, AppendBuildSettings, m_Sources, bounds);
```

```
}
```

- 캐릭터의 목적지 설정

- LTK_Human_Move_Point.Cs의 주요기능

```
// 목적지
```

```
public GameObject Target;
```

```
// 건물들의 입구를 저장하는 배열
```



```
GameObject[] Building_List;
```

(중략...)

// 건물들의 입구를 Tag명으로 찾는다.

```
Building_List = GameObject.FindGameObjectsWithTag("Target");
```

(중략...)

// 건물 목적지를 설정하는 함수

```
public void NomalState_Found_Return_GameObject()
```

```
{
```

```
    NoMalSate_RandomNum();
```

```
    Target = Building_List[choice];
```

```
}
```

//랜덤으로 목적지를 선택하는 함수

```
void NoMalSate_RandomNum()
```

```
{
```

```
    int newNum = 0;
```

```
    while (true)
```

```
    {
```

```
        //중복 검사
```

```
        if (newNum == choice)
```

```
        {
```

```
            newNum = Random.Range(0, Building_List.Length);
```

```
        }
```

```

else
{
    //건물 목적지 배열에 저장된 인덱스 번호로 넘겨준다.

    choice = newNum;

    isIn = false;

    break;
}
}
}

```

- Nomal 상태의 이동
- LTK_Human_Move.Cs의 주요기능

```
void On_Navigation()
```

```

{
    if (LTK_Point.Target != null)
    {
        //현재 캐릭터와 목적지 간의 거리를 계산

        navdis = Vector3.Distance(LTK_Point.Target.transform.position, charactor_position);

        //목적지로 캐릭터를 이동 시키는 함수

        StartCoroutine(Move_Navigation());
    }
}

```

(중략...)

//현재 캐릭터의 위치가 목적지 간의 거리가 3이하일 때

```
if (navdis < 3)
```

```
{  
  
    //캐릭터의 움직임을 멈춘다.  
  
    Agent.enabled = false;  
  
    //목적지를 재 설정한다.  
  
    isArrive = false;  
  
}  
  
(중략...)  
}  
  
IEnumerator Move_Navigation() {  
  
    //현 캐릭터의 위치와 목적지 간의 코너의 개수를 계산  
  
    NavMesh.CalculatePath(transform.position, LTK_Point.Target.transform.position,  
                           NavMesh.AllAreas, navpath);  
  
    //현 캐릭터의 위치와 목적지 간의 코너의 개수가 1개 이상 일 때까지 동작  
  
    if (navpath.corners.Length >= 1)  
    {  
  
        //가장 가까운 코너와 캐릭터의 위치의 거리차이를 계산  
  
        float temp_navdis = Vector3.Distance(navpath.corners[1], transform.position);  
  
        if (temp_navdis > 0.5f)  
        {  
  
            Agent.enabled = true;  
  
            //0.5에 가까울 때 까지 캐릭터를 코너에 가깝게 움직인다.  
  
            Agent.SetDestination(navpath.corners[1]);  
        }  
    }  
}
```

```

        TrunAngle();

    }

    else

    {

        if (navpath.corners.Length >= 2)

        {

            //가까운 코너와 캐릭터의 위치가 0.5보다 가까울 때 다음으로 가까운 코너로 목적지를
            //변경한다.

            NextCorner(navpath.corners[2]);

        }

    }

}

yield return new WaitForSeconds(0.5f);
}

```

퍼지논리

➤ 그래프 셋팅

```

public AnimationCurve Nomal_health;

public AnimationCurve Distance;

void Health_Curve_Setting()

{

    Nomal_health = new AnimationCurve(new Keyframe(20, 0), new Keyframe(100, 100));

}

```

```
void Distance_Curve_Setting()
```

```
{
```

```
    Distance = new AnimationCurve(new Keyframe(0, 0), new Keyframe(20, 100));
```

```
}
```

➤ 커브 값 계산

```
IEnumerator Curve_Distribution()
```

```
{
```

```
    float True_Health;
```

```
    float True_past_navi;
```

```
    float True_next_navi;
```

```
    //캐릭터의 체력 계산
```

```
    True_Health = Nomal_health.Evaluate(Charctor_health);
```

```
    //캐릭터와 이전 목적지의 거리계산
```

```
    True_past_navi = Distance.Evaluate(past_navi);
```

```
    //캐릭터와 다음 목적지의 거리계산
```

```
    True_next_navi = Distance.Evaluate(next_navi);
```

```
    Return_Fuzzy = (True_Health + (True_past_navi + True_next_navi) / 2) / 2;
```

```
    Health_Fuzzy = True_Health;
```

```
    Distance_Fuzzy = (True_past_navi + True_next_navi) / 2;
```

```
    Return_isRun();
```

```
    yield return new WaitForSeconds(0.5f);
```

```
}
```

➤ 역 퍼지화

```

void Return_isRun()
{
    if (Distance_Fuzzy > 75 && Health_Fuzzy > 50)
    {
        LTK_Human.isRun = true;
    }
    else
    {
        LTK_Human.isRun = false;
    }
}

```

● 차량의 이동

A* 알고리즘

➤ 차량이 이동 가능한 경로 생성

➤ LTK_Car_Road_Bake.cs의 주요기능

```

void UpdateNavMesh(bool asyncUpdate = false)
{
    //Tag가 선언된 곳 길 Object의 모음

    LTK_AI_Car_Road_Tag.Collect(ref m_Sources);

    //Agent Type[캐릭터(1) OR 차량(2)]의 인덱스 번호를 읽어 해당 Type을 읽어 변수에 저장한다.

    var AppendBuildSettings = NavMesh.GetSettingsByIndex(2);

    var bounds = QuantizedBounds();

```

```

    if (asyncUpdate)
        m_Operation = NavMeshBuilder.UpdateNavMeshDataAsync(m_NavMesh, AppendBuildSettings, m_Sources,
        bounds);

    else

        //저장된 변수와 모아진 길 Object를 Bake를 해서 경로로 인식하게 만든다.

        NavMeshBuilder.UpdateNavMeshData(m_NavMesh, AppendBuildSettings, m_Sources, bounds);
}

```

- 차량의 목적지 설정
- LTK_Human_Car_Point.Cs의 주요기능

// 목적지

```
public GameObject Target;
```

// 건물들의 입구를 저장하는 배열

```
GameObject[] Building_List;
```

// 차량이 다니는 길들을 저장하는 배열

```
GameObject[] Car_Road_List;
```

(중략...)

// 건물들의 입구를 Tag명으로 찾는다.

```
Building_List = GameObject.FindGameObjectsWithTag("Target");
```

// 차량이 다니는 길들을 Tag명으로 찾는다.

```
Car_Road_List = GameObject.FindGameObjectsWithTag("Car_AI_Path");
```

(중략...)

// 건물 목적지를 설정하는 함수

```
public void Nomal_RandomCarPath()
{
    if (Building_List != null) {
        int newNum = Random.Range(0, Building_List.Length);
        while (true)
        {
            if (newNum == choice)
            {
                newNum = Random.Range(0, Building_List.Length);
            }
            else
            {
                choice = newNum;
                CarPathNeerTarget();
                break;
            }
        }
    }
    else
    {
        Debug.Log("타겟이 필요합니다.");
    }
}
```


//선택된 건물과 가장 가까운 차도를 선택하는 작업

```
void CarPathNeerTarget()
{
    GameObject Path_Road = null;

    bool Frist_Check = true;

    float temp_navdis = 0;

    float min_nadis = 0;

    foreach (GameObject Car_Road_Path in Car_Road_List)
    {
        temp_navdis = Vector3.Distance(Building_List[choice].transform.position,
                                        Car_Road_Path.transform.position);

        if(Frist_Check)
        {
            Path_Road = Car_Road_Path;

            Frist_Check = false;

            min_nadis = temp_navdis;
        }

        if (min_nadis > temp_navdis)
        {
            min_nadis = temp_navdis;

            Path_Road = Car_Road_Path;
        }
    }
}
```

//목적지와 가장 가까운 차도를 선택해 목적지로 선택한다.

```
Target = Path_Road;
```

```
}
```

➤ Nomal 상태의 이동

➤ LTK_Car_Move.Cs의 주요기능

```
void On_Navigation()
```

```
{
```

```
    if (LTK_Car_Point.Target != null)
```

```
    {
```

```
        navdis = Vector3.Distance(LTK_Car_Point.Target.transform.position, car_position);
```

```
        //차량과 목적지 사이의 거리를 계산
```

```
        if (navdis >= 5)
```

```
        {
```

//목적지와 차량의 간격이 5 이상일 때

```
        Agent.SetDestination(LTK_Car_Point.Target.transform.position);
```

```
    }
```

```
    else
```

```
    {
```

```
        Agent.enabled = false;
```

//목적지를 재설정한다.

```
        isArrive = false;
```

```
    }
```

```
}
```

}

퍼지논리

➤ 차량간의 거리를 계산

➤ LTK_Car_Side_Mirror.cs의 주요기능

```
Front_TargetMask = LayerMask.GetMask("Vehicle_Front"); //차량의 앞면
```

```
Back_TargetMask = LayerMask.GetMask("Vehicle_Back"); //차량의 뒷면
```

//차량의 시야 각도 계산

```
Vector3 DirFromAngle(float angleInDegrees)
```

{

```
angleInDegrees += transform.eulerAngles.y;
```

```
return new Vector3(Mathf.Sin(angleInDegrees * Mathf.Deg2Rad), 0,
    Mathf.Cos(angleInDegrees * Mathf.Deg2Rad));
```

}

//차량의 거리를 재기 위한 변수 선언

```
void Find_Targets()
```

{

```
Collider[] Front_targets = Physics.OverlapSphere(car_Position, ViewDistance, Front_TargetMask);
```

```
Collider[] Back_targets = Physics.OverlapSphere(car_Position, ViewDistance, Back_TargetMask);
```

//시야 각에서부터 다른 차량의 앞면간의 거리를 담아 놓는 변수

```
Car_Front_List = new List<float>();
```

//시야 각에서부터 다른 차량의 뒷면간의 거리를 담아 놓는 변수

```
Car_Back_List = new List<float>();
```

```
Find_Collider(Front_targets, Front);

Find_Collider(Back_targets, Back);

//Debug.Log(Car_Back_List.Count);
}

//시야 각 안에 해당하는 Layer를 읽어 List에 등록한다.
void Find_Collider(Collider[] targets, int Fuzzy_List)
{
    foreach (Collider target in targets)
    {
        Vector3 target_Position = target.transform.position;

        //차로부터 타겟까지의 벡터값
        Vector3 dirToTarget = (target_Position - car_Position).normalized;

        //단위 벡터의 내적값을 비교
        if (Vector3.Dot(transform.forward, dirToTarget) > Mathf.Cos((ViewAngle / 2) * Mathf.Deg2Rad))
        {
            float distToTarget = Vector3.Distance(car_Position, target_Position);

            if (Physics.Raycast(car_Position, dirToTarget, distToTarget))
            {
                Draw_Line(target_Position, Fuzzy_List);

                //해당하는 차량의 위치를 기록한다.

                Update_Fuzzy_List(target_Position, Fuzzy_List);
            }
        }
    }
}
```

```
    }  
  
    }  
  
}
```

➤ 그래프 셋팅

```
public AnimationCurve Car_Distance;  
  
void Distance_Curve_Setting()  
{  
  
    Car_Distance = new AnimationCurve(new Keyframe(0, 100), new Keyframe(15, 0));  
  
}
```

➤ 커브값 계산

```
void Curve_Distribution()  
{  
  
    float Avg_Distance;  
  
    float Back_Distance = 0;  
  
    float Front_Distance = 0;  
  
    LTK_Side.Car_Front_List.Sort();  
  
    LTK_Side.Car_Back_List.Sort();  
  
    if (LTK_Side.Car_Front_List.Count > 0)  
    {  
  
        Front_Distance = Neer_Distance(LTK_Side.Car_Front_List[Close]);  
  
    }  
  
    if (LTK_Side.Car_Back_List.Count > 0)  
    {  
  
        Back_Distance = Neer_Distance(LTK_Side.Car_Back_List[Close]);  
  
    }  
  
}
```

```
}

Avg_Distance = (Front_Distance + Back_Distance) / 2;

return_car_fuzzy_speed = Return_Fuzzy_Speed(Avg_Distance, Front_Distance, Back_Distance);
}
```

➤ 역퍼지화

```
float Return_Fuzzy_Speed(float Avg, float Front, float Back)
```

```
{

    float temp_speed = 1;

    if (Back >= 70)

    {

        return temp_speed = 0;

    }

    else if (Back < 70 && Back >= 50)

    {

        return temp_speed = 4;

    }

    else if (Back < 50 && Back >= 40)

    {

        return temp_speed = 2;

    }

    if (Front >= 60)

    {

        return temp_speed = 4;

    }

}
```

```
}  
  
else if(Front < 60 && Front >= 40)  
{  
    return temp_speed = 2;  
}  
  
if(Avg >= 40)  
{  
    return temp_speed = 2;  
}  
  
return temp_speed;  
}
```

➤ 건물배치

```
public GameObject M_Obj;

public GameObject m_gameobj;

[SerializeField]

private LayerMask buildableSurfacesLayer;

private Vector3 buildPos;

private GameObject currentTemplateBlock;

[SerializeField]

private GameObject blockTemplatePrefab;

[SerializeField]

private GameObject blockPrefab;

[SerializeField]

private Material templateMaterial;

IEnumerator BuildObj(){

    if (M_Obj != null)

    {

        blockTemplatePrefab = M_Obj;

        blockPrefab = M_Obj;

    }

    if (Input.GetKeyDown("e"))

    {

        buildModeOn = !buildModeOn;

    }

}
```



```
if (buildModeOn)
{
    RaycastHit buildPosHit;

    if (Physics.Raycast(m_camera.ScreenPointToRay(Input.mousePosition), out
        buildPosHit, 1000, buildableSurfacesLayer))
    {
        Vector3 point = buildPosHit.point;

        if (blockTemplatePrefab != null)
        {
            if (blockTemplatePrefab.GetComponent<MeshRenderer>() != null)

                buildPos = new Vector3(Mathf.Round(point.x), Mathf.Round(point.y) +
                    ((blockTemplatePrefab.GetComponent<MeshRenderer>().bounds.size.y) / 2),
                    Mathf.Round(point.z));
        }

        if (canBuild && currentTemplateBlock != null)
        {
            currentTemplateBlock.transform.position = buildPos;

            if (Input.GetMouseButtonDown(0))
            {
                PlaceBlock();
            }
        }

        yield return new WaitForEndOfFrame();
    }
}
```

```
}  
  
private void PlaceBlock()  
{  
    GameObject newBlock;  
  
    newBlock = Instantiate(blockPrefab, buildPos, Quaternion.identity);  
  
    newBlock.transform.rotation = currentTemplateBlock.transform.rotation;  
}
```

➤ 세이브/로드

```
public class SaveInfo
{
    public string Pos;
    public string Rot;
    public string Tag;
    public int ID;

    public SaveInfo(string pos, string rot, string tag, int id)
    {
        ID = id;
        Pos = pos;
        Rot = rot;
        Tag = tag;
    }
}
```

```
GameObject[] background;
GameObject[] building;
GameObject[] road;
GameObject[] structure;
GameObject[] nature;
GameObject[] vehicle;
GameObject[] character;
```

```
GameObject[] other;

public List<SaveInfo> save_list;

public SaveInfo SI;

void Start()
{
    save_list = new List<SaveInfo>();

    savepath = "C:/Users/bit-user/Desktop/HoloCity/sname.json";
    loadpath = "C:/Users/bit-user/Desktop/HoloCity/lname.json";
}

public void Save_Btn()
{
    save_list = new List<SaveInfo>();

    StartCoroutine(SaveData());
}

public void Load_Btn()
{
    StartCoroutine(LoadData());
}

IEnumerator LoadData()
{
    Load_Destroy();

    string Jsonstring = File.ReadAllText(loadpath);
```

```
//Debug.Log(Jsonstring);

JsonData UserData = JsonMapper.ToObject(Jsonstring);

CreateObj(UserData);

yield return null;

}

private void CreateObj(JsonData name)

{

    //Debug.Log("불러오기");

    for (int i = 0; i < name.Count; i++)

    {

        int tmpId = int.Parse(name[i]["ID"].ToString());

        string tmpPos = name[i]["Pos"].ToString();

        string tmpRot = name[i]["Rot"].ToString();

        string tmpTag = name[i]["Tag"].ToString();

        string[] Pos = tmpPos.Split('/');

        string[] Rot = tmpRot.Split('/');

        Vector3 obj_Pos = new Vector3(float.Parse(Pos[0]), float.Parse(Pos[1]),
                                         float.Parse(Pos[2]));

        Vector3 obj_Rot = new Vector3(float.Parse(Rot[0]), float.Parse(Rot[1]),
                                         float.Parse(Rot[2]));

        if (tmpTag == "GROUND")

        {
```

```
        for (int j = 0; j < item.Background_ItemDB.Count; j++)
        {
            if (tmpId == item.Background_ItemDB[j].itemID)
            {
                GameObject tmp = item.Background_ItemDB[j].itemModel;

                Instantiate(tmp, obj_Pos, Quaternion.Euler(obj_Rot));
            }
        }
    }

IEnumerator SaveData()
{
    //Debug.Log("저장됨");

    background = GameObject.FindGameObjectsWithTag("GROUND");

    if (background != null)
    {
        foreach (GameObject info in background)
        {
            string pos = info.transform.position.x + "/" + info.transform.position.x + "/" +
                info.transform.position.z;

            string rot = info.transform.rotation.eulerAngles.x + "/" +
                info.transform.rotation.eulerAngles.y + "/" +
                info.transform.rotation.eulerAngles.z;

            string tag = info.gameObject.tag;

            int id = info.gameObject.GetComponent<LTK_Item_Info>().Item_Id;
```

```
        SI = new SaveInfo(pos, rot, tag, id);

        save_list.Add(SI);
    }
}

JsonData UserJson = JsonMapper.ToJson(save_list);
File.WriteAllText(savepath, UserJson.ToString());

yield return null;
}
```

➤ 립모션 이동

```
Controller m_controller;

bool rotatemove = false;

bool goforward = false;

GameObject m_objCamera;

GameObject m_objTarget;

Vector3 m_vCameraRotation;

Vector3 relativePos;

Frame m_CurrentFrame;
```

```
Frame m_PreviousFrame;

void Start()
{
    m_controller = new Controller();
    m_objCamera = GameObject.Find("Main Camera");
    m_vCameraRotation = new Vector3(0, 0, 0);
    relativePos = new Vector3(0, 0, 0);
}

void Update()
{
    // 현재 및 이전 프레임 로드
    Frame fCurFrame = m_controller.Frame();
    m_CurrentFrame = fCurFrame;
    Frame fPreFrame = m_controller.Frame(1);
    m_PreviousFrame = fPreFrame;

    StartCoroutine(ControlLeftHands());
    StartCoroutine(ControlRightHands());
}

IEnumerator ControlLeftHands()
{
    Frame fCurFrame = m_CurrentFrame;
```



```
foreach (Hand h in fCurFrame.Hands)
{
    if (h.IsLeft)
    {
        if (h.Finger(h.Fingers[1].Id).IsExtended == false &&
            h.Finger(h.Fingers[2].Id).IsExtended == false &&
            h.Finger(h.Fingers[3].Id).IsExtended == false &&
            h.Finger(h.Fingers[4].Id).IsExtended == false)
            rotatemove = true;

        else
        {
            rotatemove = false;

            goforward = false;
        }
    }
}

yield return new WaitForEndOfFrame();
}

IEnumerator ControlrightHands()
{
    Frame fCurFrame = m_CurrentFrame;

    Frame fPreFrame = m_PreviousFrame;

    // 현재 및 이전 프레임에 들어오는 첫번째 손 찾기
```

```
float curx = 0;

float prex = 0;

float cury = 0;

float prey = 0;

//m_vCameraRotation = new Vector3(0, 0, 0);

foreach (Hand h in fPreFrame.Hands)
{
    if (h.IsRight)
    {
        prex = h.PalmPosition.x;

        prey = h.PalmPosition.y;
    }
}

foreach (Hand h in fCurFrame.Hands)
{
    if (h.IsRight)
    {
        curx = h.PalmPosition.x;

        cury = h.PalmPosition.y;

        if (rotatemove)
        {
```

```
//카메라 회전

if (h.Finger(h.Fingers[0].Id).IsExtended == true &&
    h.Finger(h.Fingers[1].Id).IsExtended == true &&
    h.Finger(h.Fingers[2].Id).IsExtended == true)
{
    if (h.PalmNormal.Roll < -1)
    {
        if (h.PalmVelocity.x > 500)
        {
            rightleftmove += 5f;

            m_objCamera.transform.rotation = Quaternion.Euler(updownmove,
                rightleftmove, 0);
        }
        else if (h.PalmVelocity.x < -500)
        {
            rightleftmove -= 5f;

            m_objCamera.transform.rotation = Quaternion.Euler(updownmove,
                rightleftmove, 0);
        }
    }
}

//전진

if (h.Finger(h.Fingers[0].Id).IsExtended == false &&
    h.Finger(h.Fingers[1].Id).IsExtended == true &&
```

```
        h.Finger(h.Fingers[2].Id).IsExtended == false &&
        h.Finger(h.Fingers[3].Id).IsExtended == false)

        goforward = true;

    else

        goforward = false;

    }

    if (goforward)

        m_objCamera.transform.Translate(Vector3.forward * 0.1f);

    if (goUp)

        m_objCamera.transform.Translate(Vector3.up * 0.1f);

    yield return new WaitForEndOfFrame();
}
```

➤ 립모션 UI 생성

```
public class OpenUI : MonoBehaviour {

    Controller m_controller;

    bool ishandUI = false;

    public GameObject m_handUI;

    GameObject CEL;

    GameObject CDL;

    GameObject cbutton;
```

```
Frame m_CurrentFrame;

Frame m_PreviousFrame;

void Start () {

    m_controller = new Controller();

}

void Update () {

    Frame fCurFrame = m_controller.Frame();

    m_CurrentFrame = fCurFrame;

    Frame fPreFrame = m_controller.Frame(1);

    m_PreviousFrame = fPreFrame;

    StartCoroutine(ControlleftHands());

    CEL = GameObject.Find("Canvas_Environmental_leap2");

    CDL = GameObject.Find("Canvas_Disaster_leap");

    cbutton = GameObject.Find("Canvasbtn");

}

IEnumerator ControlleftHands()

{
```

```
Frame fCurFrame = m_CurrentFrame;

foreach (Hand h in fCurFrame.Hands)
{
    if (h.IsLeft)
    {
        if ((h.PalmNormal.Roll > 1.8f && h.PalmNormal.Roll < 3.5f) || (h.PalmNormal.Roll
< -2.8f))
        {
            CEL.gameObject.GetComponent<Canvas>().enabled = true;
            CDL.gameObject.GetComponent<Canvas>().enabled = true;
            cbutton.gameObject.GetComponent<Canvas>().enabled = true;
        }
        else
        {
            CEL.gameObject.GetComponent<Canvas>().enabled = false;
            CDL.gameObject.GetComponent<Canvas>().enabled = false;
            cbutton.gameObject.GetComponent<Canvas>().enabled = false;
        }
    }
}

yield return new WaitForEndOfFrame();
}
```

➤ 립모션클릭

```
public class csLeapMenuClick : MonoBehaviour {

    Controller controller;

    //날씨
    public GameObject rain;
    public GameObject snow;
    public GameObject exit;

    void Start () {
        controller = new Controller();
    }

    void Update () {
        Frame frame = controller.Frame();
    }

    void OnTriggerEnter(Collider col)
    {
        //재해
        if (col.gameObject.name == "ear thquake")
        {
```

```
        ctm.Mode = "Earthquake";
;    }

    if(col.gameObject.name == "tornado")
    {
        ctm.Mode = "Tornado";
    }

    if (col.gameObject.name == "firework")
    {
        ctm.Mode = "FireBuilding";
    }

    if (col.gameObject.name == "reset")
    {
        ctm.Mode = "Normal";
    }

    //PC모드로 전환
    if (col.gameObject.name == "turnoff")
    {
        scr.LeaptoPc();
    }

    //날씨
    if (col.gameObject.name == "rain")
    {
```



```
sbc.Weatherchoice(rain);  
  
sbc.Cloudsky();  
  
}  
  
if (col.gameObject.name == "snow")  
{  
  
    sbc.Weatherchoice(snow);  
  
    sbc.Cloudsky();  
  
}  
  
if (col.gameObject.name == "exit")  
{  
  
    sbc.Weatherchoice(exit);  
  
}  
  
}  
  
}
```

➤ 디스플레이 멀티출력

```
using UnityEngine;
using System.Collections;

public class DisplayScript : MonoBehaviour
{
    void Start()
    {
        for(int i = 0; i<Display.displays.Length; i++)
        {
            Display.displays[i].Activate(1024, 768, 60);
        }
    }
}
```

➤ 스카이박스변경

```
public class csSkyboxChange : MonoBehaviour {

    public Material Sunny;

    public Material Cloud;

    GameObject SunLight;
```

```
GameObject Save_Weather;

GameObject G_Rain;

public Light lt;

public Slider slider;

public Text mytext;

void Start () {

    SunLight = GameObject.Find("SunLight");

    lt = SunLight.GetComponent<Light>();

    RenderSettings.skybox = Sunny;

    slider = GameObject.Find("Slider").GetComponent<Slider>();

    mytext = GameObject.Find("btnText").GetComponent<Text>();

    RenderSettings.skybox.SetFloat("_Exposure", 1f);

}

void Update()

{

    G_Rain = GameObject.Find("Rain(Clone)");

    slider = GameObject.Find("Slider").GetComponent<Slider>();

    mytext = GameObject.Find("btnText").GetComponent<Text>();

}

public void SliderValueChange()
```

```
{

    float Sval = slider.value;

    if (Sval < 0.48)
    {

        RenderSettings.skybox.SetFloat("_Exposure", 1f - (1f - (Sval + 0.5f)));

        SunLight.transform.rotation = Quaternion.Euler(90 - ((1f - Sval) * 100f),
                                                         -90, -90);

    }

}

public void Weatherchoice(GameObject Choice)
{

    if (Choice.tag == "Rain")
    {

        if (G_Rain == null)
        {

            Instantiate(Weather.arr[0], position, Quaternion.Euler(-90, 0, 0));

            RenderSettings.skybox = Cloud;

            Debug.Log("button Clicked!");

            lt.intensity = 0.75f;

            LampOn = true;

            foreach (GameObject Um in Umbrella)
            {
```

```
        Um.GetComponent<MeshRenderer>().enabled = true;

    }

}

}

}

public void SunnySky()
{
    RenderSettings.skybox = Sunny;

    Debug.Log("button Clicked!");

    lt.intensity = 1.5f;
}
}
```

➤ 재해

```
//화재

IEnumerator BuildingFire()
{
    isstart = false;

    RaycastHit burningHit;

    if (pcselect)
    {
        if (Physics.Raycast(m_camera.ScreenPointToRay(Input.mousePosition), out
burningHit, 1000, MoveObjectLayer))
        {
            Debug.Log("클릭전");

            if (Input.GetMouseButtonDown(0))
            {
                Debug.Log("클릭후");

                if (burningHit.transform.gameObject.tag == "Buildings")
                {
                    TargetBuilding = burningHit.transform.gameObject;

                    Debug.Log("TB : " + TargetBuilding.name);

                    StartCoroutine(BurningBuilding());

                }
            }
        }

        Debug.DrawRay(m_camera.transform.position, m_camera.transform.forward * 1000,
```

```
Color.red);  
  
    }  
  
    yield return new WaitForEndOfFrame();  
  
    isstart = true;  
}  
  
IEnumerator BurningBuilding()  
{  
    GameObject fb = TargetBuilding;  
  
    Vector3 fb_obstacle = fb.GetComponent<NavMeshObstacle>().size;  
  
    fb.GetComponent<NavMeshObstacle>().size *= 2;  
  
    int child_cnt = fb.transform.childCount;  
  
    for (int i = 0; i < child_cnt; i++)  
    {  
        if (fb.transform.GetChild(i).tag == "Target")  
        {  
            Debug.Log(fb.transform.GetChild(i).name);  
  
            fb.transform.GetChild(i).tag = "noneTarget";  
        }  
    }  
  
    while (makefire)  
    {  
        fb.tag = "Burning";  
    }  
}
```

```
        if (smokeparticle != null && fireparticle != null)
        {
            GameObject fireeffect = Instantiate(fireparticle) as GameObject;
            fireeffect.transform.position = fb.transform.position;
            fireeffect.transform.parent = transform;
            Destroy(fireeffect, 2f);

            GameObject smokeeffect = Instantiate(smokeparticle) as GameObject;
            smokeeffect.transform.position = fb.transform.position;
            smokeeffect.transform.parent = transform;
            Destroy(smokeeffect, 2f);
        }

        yield return new WaitForSeconds(1f);
    }

    fb.tag = "Buildings";

    fb.GetComponent<NavMeshObstacle>().size = fb_obstacle;

    for (int i = 0; i < child_cnt; i++)
    {
        if (fb.transform.GetChild(i).tag == "noneTarget")
        {
            fb.transform.GetChild(i).tag = "Target";
        }
    }
}
```



```
//지진

IEnumerator EarthquakeDamage()
{
    if (LTK_Save.OnOff)
    {
        isstart = false;

        Buildings = GameObject.FindGameObjectsWithTag("Buildings");

        foreach (GameObject bd in Buildings)
        {
            bd.transform.Translate(new Vector3(0, speedY, 0), Space.World);

            bd.transform.Translate(new Vector3(0, 0, 0), Space.World);
        }

        if (pcselect)
        {
            obj_Camera.transform.Translate(new Vector3(0, speedY, 0), Space.World);

            obj_Camera.transform.Translate(new Vector3(0, 0, 0), Space.World);
        }

        yield return new WaitForSeconds(0.01f);

        isstart = true;
    }
}
```

```
//태풍

IEnumerator Tornado()
{
    isstart = false;

    if (Realtornado == null)

        Realtornado = Instantiate(tornado, new Vector3(301f, 0, 310f), Quaternion.identity);

    yield return new WaitForSeconds(0.01f);

    isstart = true;
}

//유성

IEnumerator MeteorStrike()
{
    isstart = false;

    RaycastHit Meteor_target;

    if(pcselect)
    {
        if(Physics.Raycast(m_camera.ScreenPointToRay( Input.mousePosition), out
Meteor_target,1000, MeteorClickLayer))
        {
            Vector3 Point = Meteor_target.point;

            if ( Input.GetMouseButtonDown(0))
            {
```

```
        if (EventSystem.current.IsPointerOverGameObject() == false)

            Instantiate(Meteor_swarm, Point, Quaternion.identity);

    }

}

}

isstart = true;

yield return null;

}
```

- 재해메뉴 중 건물내부로 썬 이동 및 사람AI의 위치선정과 화재 발생

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class csEventChange : MonoBehaviour {
    public string mode;
    GameObject[] characters;
    GameObject CheckTemp;
    GameObject[] Fires;
    GameObject Fire;
    AudioSource sound;

    // Use this for initialization
    void Start () {
        mode = "Normal";
        characters = GameObject.FindGameObjectsWithTag("Character");
        CheckTemp = GameObject.Find("checktemp");
        CheckTemp.GetComponent<cschecktemp>().temp = true;
        Fire = GameObject.Find("FIRE");
        Fires = new GameObject[Fire.transform.childCount];
        for(int i = 0; i<Fire.transform.childCount; i++)
        {
            Fires[i] = Fire.transform.GetChild(i).gameObject;
        }
        sound = GetComponent<AudioSource>();
    }

    public void setnormal()
    {
        mode = "Normal";
        sound.enabled = false;
        foreach (GameObject ch in characters)
        {
```

```
        ch.transform.position = ch.GetComponent<csCheckPriority>().firstpos;
        ch.transform.rotation = ch.GetComponent<csCheckPriority>().firstrot;
        ch.GetComponent<csCheckPriority>().reset = true;
        ch.SetActive(true);
    }
    foreach(GameObject f in Fires)
    {
        f.SetActive(false);
    }
}

public void setEvent()
{
    mode = "Event";
}

public void setEscape()
{
    mode = "Escape";
    sound.enabled = true;
    StartCoroutine(BuildingInsideFire());
}

IEnumerator BuildingInsideFire()
{
    foreach(GameObject f in Fires)
    {
        f.SetActive(true);

        yield return new WaitForSeconds(0.6f);
    }
}

public void ChangeScene()
{

```

```
SceneManager.LoadScene("02_3char_ING");  
    }  
}
```

- 건물내부 상황에서의 사람AI의 일반상황 움직임과 대피상황 움직임

```
using System.Collections;  
  
using System.Collections.Generic;  
  
using UnityEngine;  
  
public class csWayPointMoveCtrl3 : MonoBehaviour {  
  
    public float speed = 1f;  
  
    public float damping = 3f;  
  
    float presentspeed = 0;  
  
    bool pause = false;  
  
    bool continues = true;  
  
    Transform tr;  
  
    Transform[] points;  
  
    Transform[] ESpoints;
```

```
Animator ani;

csCheckPriority CP;

string mode;

int nextIdx = 1;

int preIdx;

bool isfirst = false;

// Use this for initialization

void Start()

{

    tr = GetComponent<Transform>();

    //tr = transform;

    points = GameObject.Find("WayPointGroup3").GetComponentsInChildren<Transform>();

    ESpoints =
GameObject.Find("WayPointGroup3toEscape").GetComponentsInChildren<Transform>();

    mode = GameObject.Find("UIEvent").GetComponent<csEventChange>().mode;

    CP = GetComponent<csCheckPriority>();

    ani = GetComponent<Animator>();

    preIdx = nextIdx;

    presentspeed = speed;
```

```
}

// Update is called once per frame

void Update()
{
    ani.SetFloat("Speed", speed);

    mode = GameObject.Find("UIEvent").GetComponent<csEventChange>().mode;

    if (CP.reset)
    {
        nextIdx = Random.Range(1, points.Length);

        CP.reset = false;
    }

    if (mode != "Escape")
        StartCoroutine(MoveWayPoint());

    else
        StartCoroutine(EscapeMoveWayPoint());
}
```



```
IEnumerator MoveWayPoint()
{
    if (continues)
    {
        if (pause)
        {
            continues = false;

            pause = false;

            speed = 0;

            yield return new WaitForSeconds(1);

            speed = presentspeed;

            continues = true;
        }

        Vector3 dir = points[nextIdx].position - tr.position;

        Quaternion rot = Quaternion.LookRotation(dir);

        tr.rotation = Quaternion.Slerp(tr.rotation, rot, Time.deltaTime * damping);

        tr.Translate(Vector3.forward * Time.deltaTime * speed);

        yield return null;
    }
}
```

```
}

IEnumerator EscapeMoveWayPoint()

{

    if (continues)

    {

        if (!isfirst)

        {

            float min = 10000;

            for (int i = 0; i < ESpoints.Length; i++)

            {

                if (Vector3.Distance(ESpoints[i].position, tr.position) < min &&
ESpoints[i].gameObject.name == "EsPoint")

                {

                    min = Vector3.Distance(ESpoints[i].position, tr.position);

                    nextIdx = i;

                }

            }

            isfirst = true;

        }

    }

}
```

```
        if (pause)
        {
            continues = false;

            pause = false;

            speed = 0;

            yield return new WaitForSeconds(1);

            speed = presentspeed;

            continues = true;
        }

        Vector3 dir = ESpoints[nextIdx].position - tr.position;

        Quaternion rot = Quaternion.LookRotation(dir);

        tr.rotation = Quaternion.Slerp(tr.rotation, rot, Time.deltaTime * damping);

        tr.Translate(Vector3.forward * Time.deltaTime * speed);

        yield return null;
    }
}

void OnTriggerEnter(Collider coll)
```

```
{

    if (mode == "Normal")

    {

        isfirst = false;

        speed = presentspeed;

        if (coll.CompareTag("WAY_POINT"))

        {

            preIdx = nextIdx;

            int temp = Random.Range(1, points.Length);

            while (temp == preIdx)

                temp = Random.Range(1, points.Length);

            nextIdx = temp;

        }

        if (coll.CompareTag("Others"))

        {

            nextIdx = preIdx;

        }

    }

    else if (mode == "Event")
```

```
{

    isfirst = false;

    if (coll.CompareTag("WAY_POINT"))

    {

        nextIdx = (++nextIdx >= points.Length) ? 1 : nextIdx;

    }

}

else if (mode == "Escape")

{

    speed = 6;

    Debug.Log(ESpoints.Length);

    if (coll.CompareTag("WAY_POINT_ESCAPE"))

    {

        if (nextIdx + 1 >= ESpoints.Length)

            nextIdx = Random.Range(1, points.Length);

        else

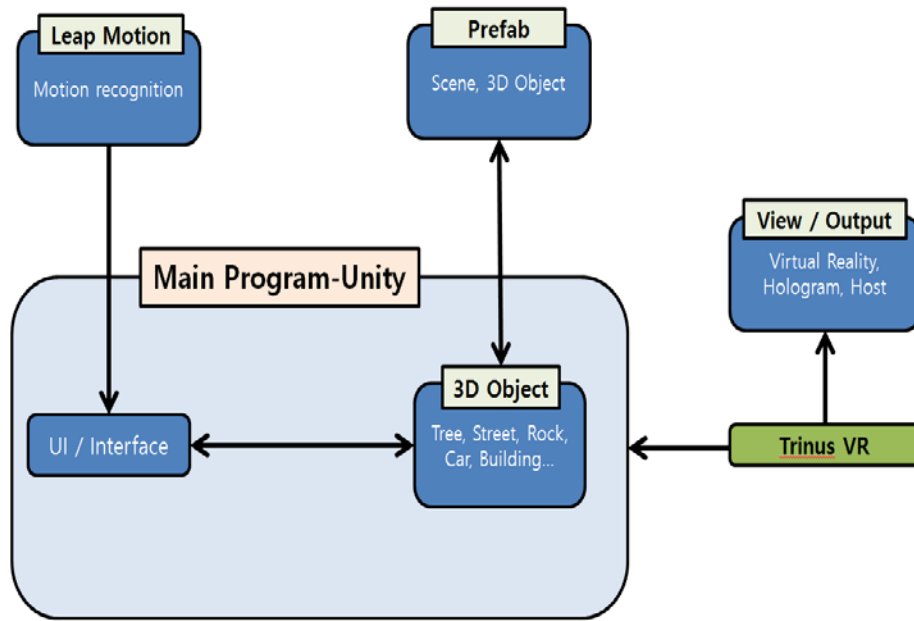
            nextIdx = ++nextIdx;

    }

}
```

```
if (coll.tag == "Character")  
  
    {  
  
        int temppriority = coll.gameObject.GetComponent<csCheckPriority>().Priority;  
  
        if (temppriority < this.GetComponent<csCheckPriority>().Priority)  
  
            pause = true;  
  
    }  
  
}  
  
}
```

III. 프로젝트 전체 아키텍처

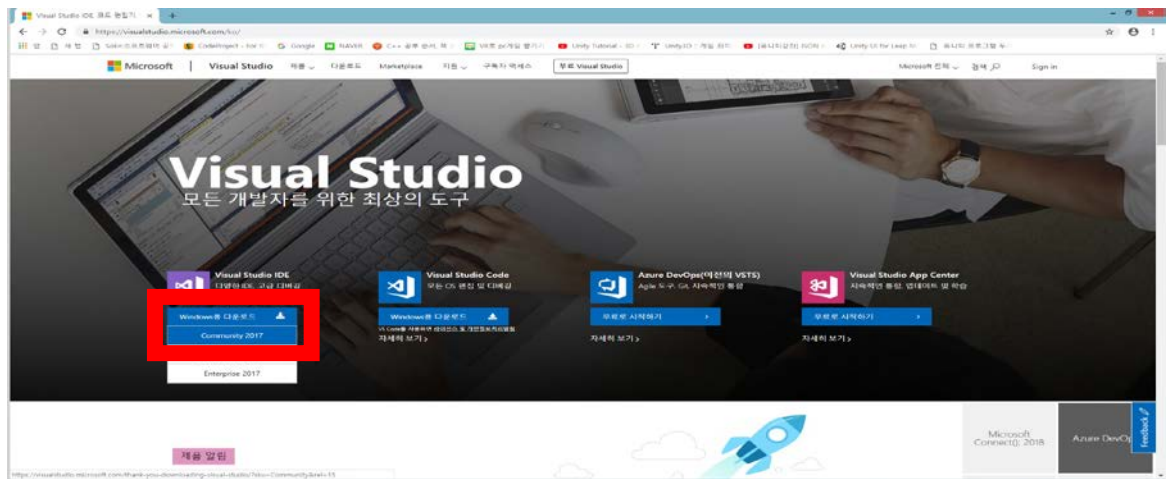


5. 사용자 매뉴얼

A. 개발환경설치

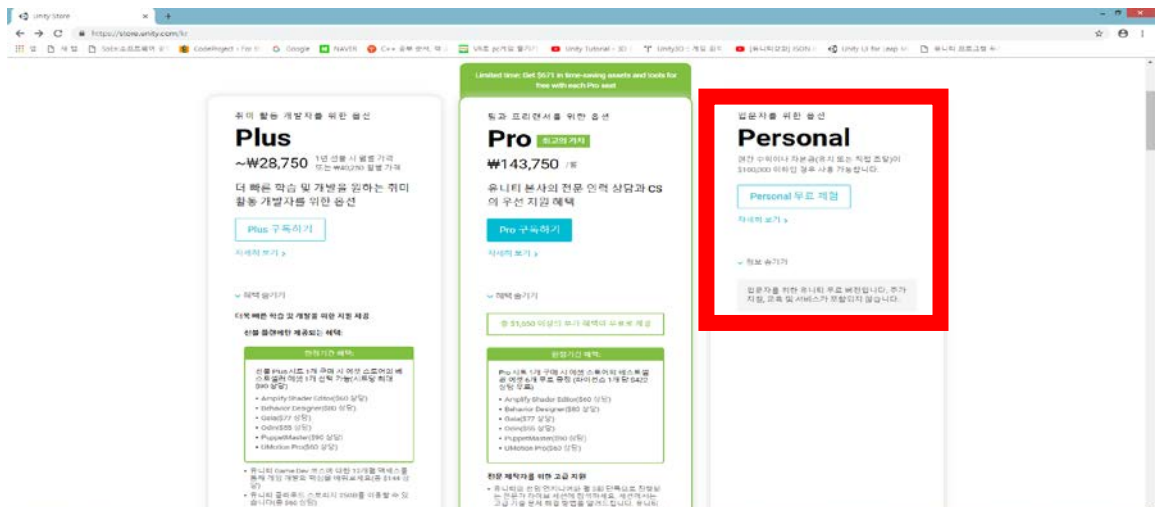
I. Visual Studio2017 설치

✓ (<https://visualstudio.microsoft.com/ko/>)접속 -> Windows용-Community2017 다운로드 후 설치



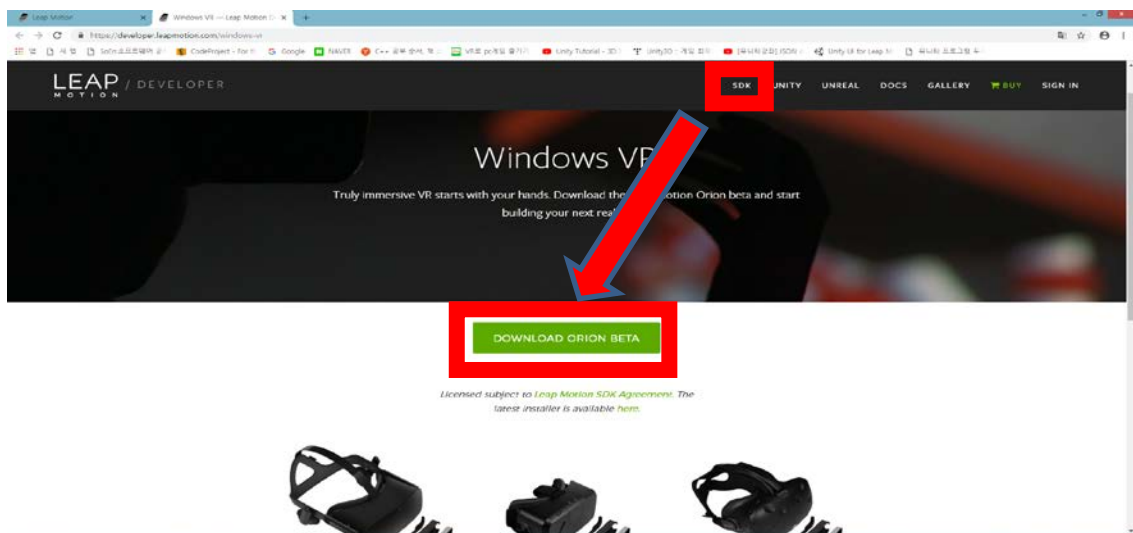
II. Unity 설치

✓ (<https://store.unity.com/kr/>)접속 -> Personal 다운로드 후 설치

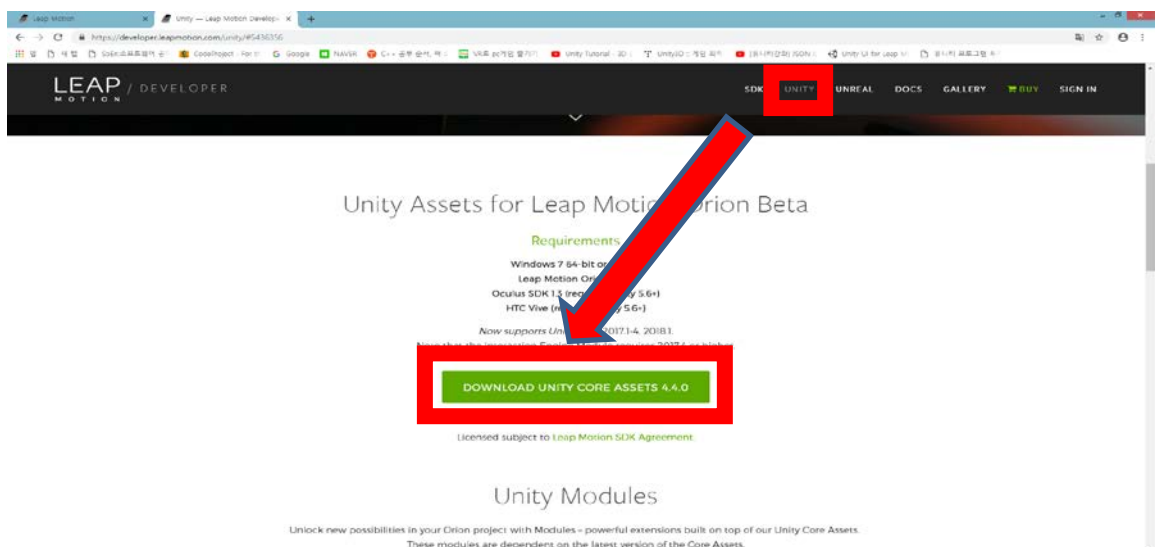


III. LeapMotion

✓ (<https://www.leapmotion.com/>) 접속 -> SDK -> ORION BETA 다운로드 후 설치

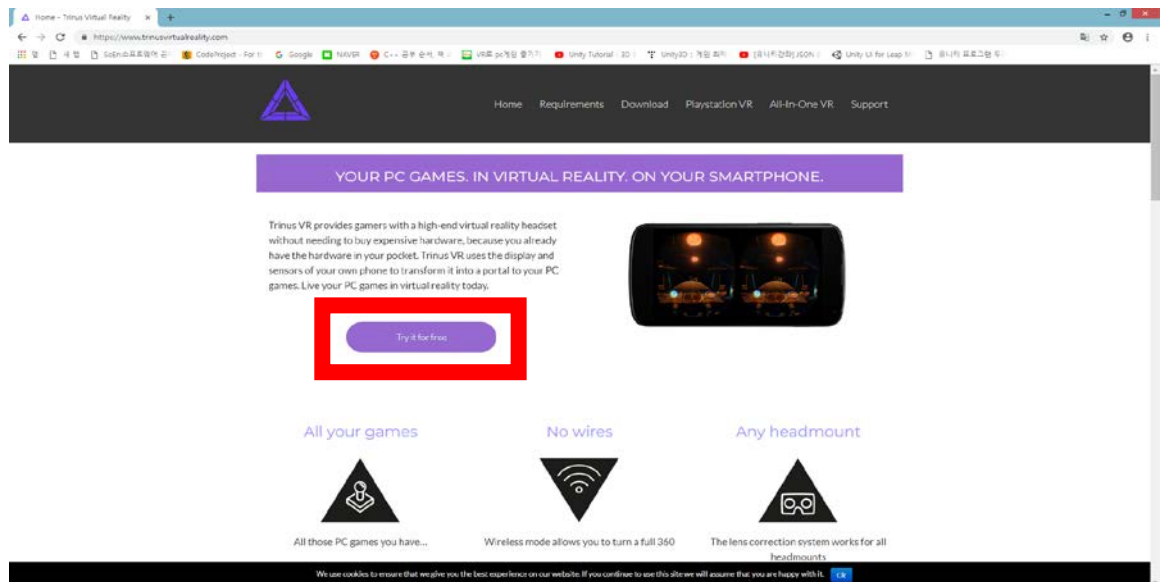


✓ UNITY -> UNITY CORE ASSETS 4.4.0 다운로드 -> Unity에 Import한다.

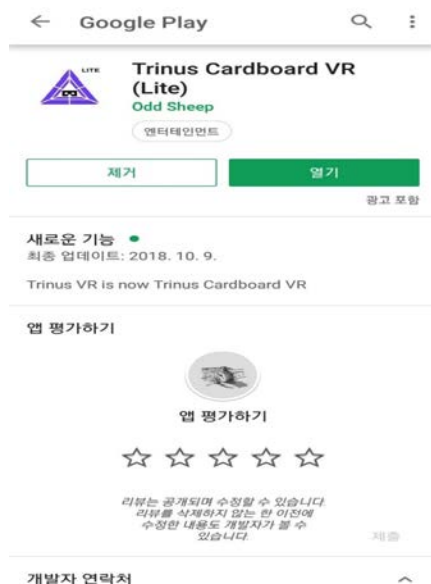


IV. Trinus VR 설치

✓ (<https://www.trinusvirtualreality.com/>) 접속 -> 다운로드 후 설치



✓ (모바일 기기)구글 플레이 스토어 -> Trinus VR검색 -> 다운로드



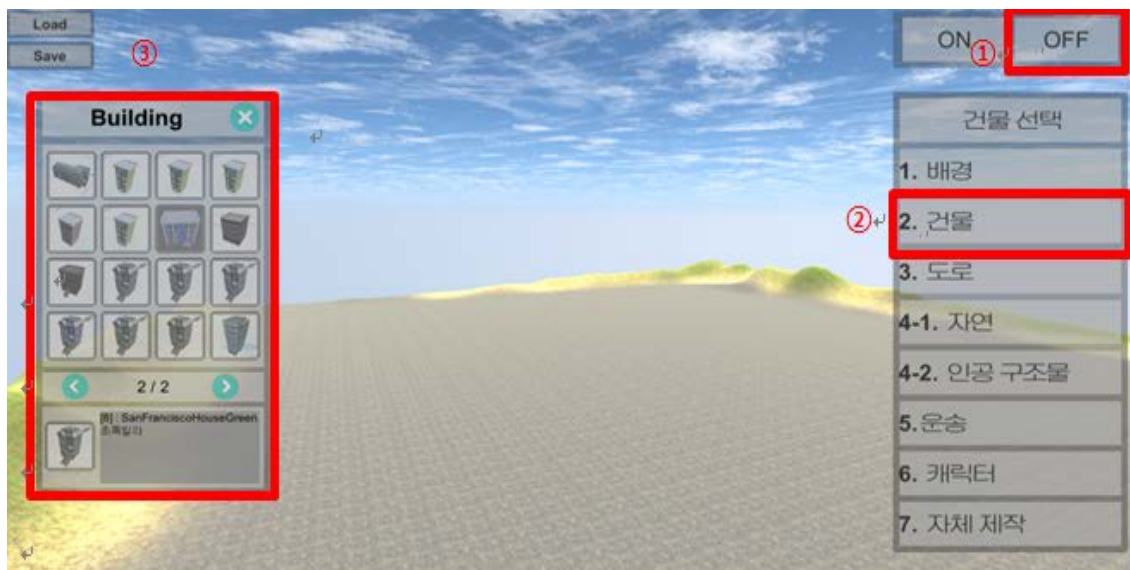
B. 조작법

사용자 키보드 매뉴얼(PC모드)	
W / ↑	앞으로 이동
S / ↓	뒤로 이동
A / ←	좌로 이동
D / →	우로 이동
E	Object 설치 미리보기 ON/OFF
F	Mouse회전 잠금 ON/OFF
R	Object 설치 미리보기 시 물체 회전
Mouse 회전	시점 이동
Mouse Left Click	Object 선택 / UI 클릭
Mouse Right Click	Object 정보UI ON/OFF

사용자 Hand Motion 매뉴얼(LeapMotion모드)	
좌 : 주먹, 우 : 검지	앞으로 이동
좌 : 주먹, 우 : 검지, 중지	뒤로 이동
좌 : 주먹, 우 : 엄지	좌로 이동
좌 : 주먹, 우 : 새끼	우로 이동
좌 : 주먹, 우 : 엄지, 검지	위로 이동
좌 : 주먹, 우 : 엄지, 검지, 중지	아래로 이동
좌 : 주먹, 우 : 상하좌우 이동	상하좌우로 시점 이동
우 : 검지로 UI 터치	UI 클릭

C. 프로그램 시뮬레이션

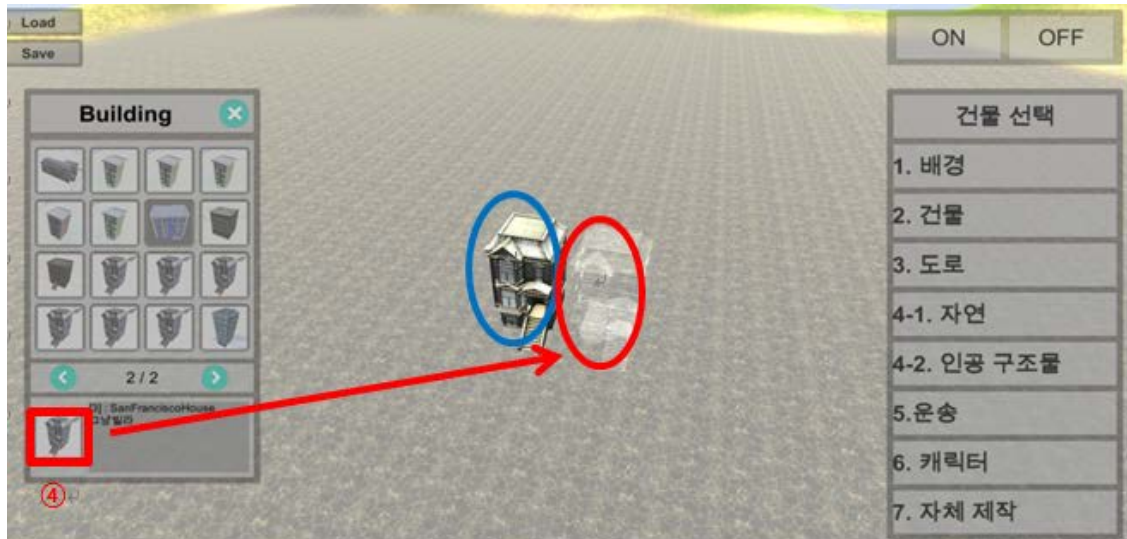
i. 시뮬레이션 OFF 모드



① 설치/배치가 가능한 OFF모드와 AI의 동적 움직임과 상황변화를 볼 수 있는 ON모드가 선택 가능하다. OFF모드는 세밀한 조정이 필요하기 때문에 PC모드만 사용 가능하다.

② 우측 Object 메뉴 카테고리들을 선택할 수 있는 UI이다. 카테고리별로 다르게 Database화 하여 리스트에 추가 되어있어 선택 가능하다.

③ 카테고리를 선택하면 해당 카테고리UI가 나타나며 설치하고자 하는 Object를 설치할 수 있다.



④ 해당 칸을 선택하면 반투명의 미리보기가 생성되어 설치하고자 하는 장소에 배치 시킬 수 있으며 마우스클릭으로 설치를 확정시키면 원래의 색으로 돌아온다. 이미 설치된 Object는 마우스클릭으로 선택하여 위치를 마우스 포인트의 위치 움직임으로 이동시킬 수 있다.

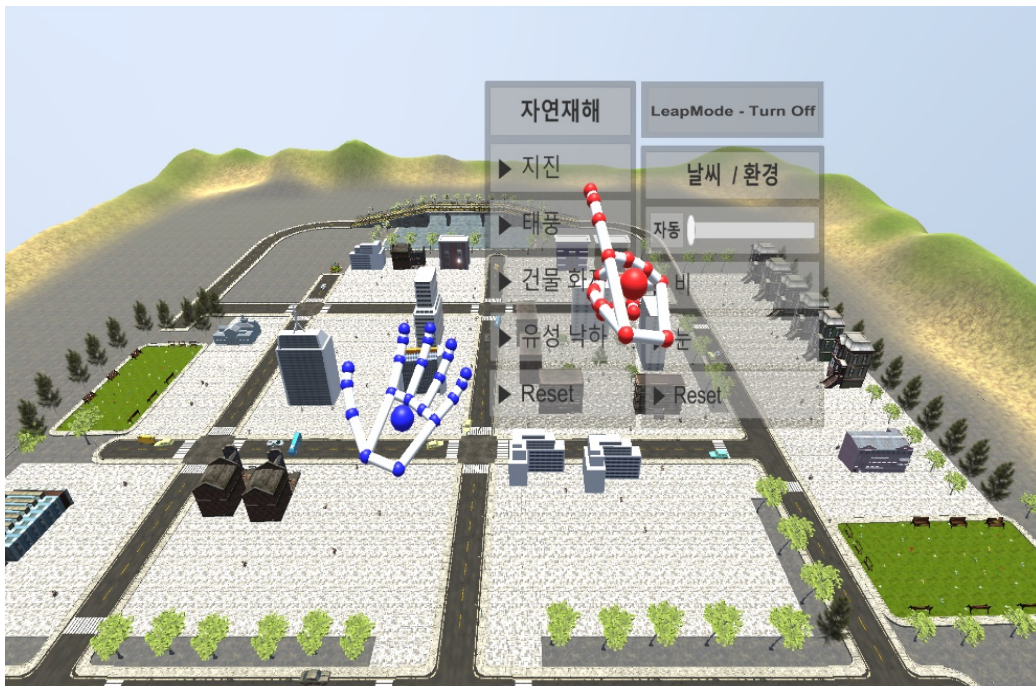
ii. 시뮬레이션 ON 모드

가) PC모드

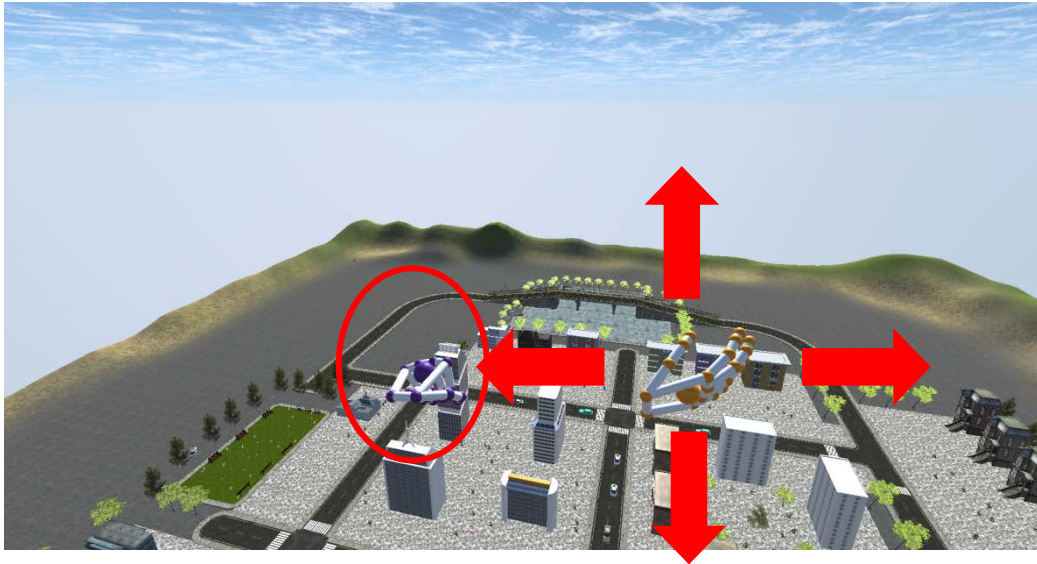


- ① ON 버튼을 누르게 되면 AI의 동적 움직임 및 상황변화 요소를 실행시킬 수 있다.
- ② LeapMotion모드로 전환 할 수 있는 버튼으로 손 모션으로 진행하는 모드로 전환가능
- ③ 상황변화요소 메뉴(지진, 태풍, 건물 화재, 건물화재 내부로 전환, 유성낙하)로 현재 상황을 변화시킬 수 있고 그에 따른 상황이 변화하며 사람AI와 차량AI가 해당 상황에 맞는 행동을 자동적으로 취하게 된다.
- ④ 환경변화요소 메뉴(조도 변화, 날씨)로 슬라이드 바를 통해 조도를 조정하여 이에따른 조명이 자동적으로 변화하며 날씨 변화로 인하여 사람AI의 모션이 변화가 자동적으로 생기게 된다.
- ⑤ 상황변화요소의 건물화재 Tab에서 건물 내부로 들어갈 수 있는 메뉴이며 건물화재Tab을 클릭할 시에만 나타난다. 클릭하게 되면 건물내부로 씬 전환이 된다.

나) LeapMotion 모드



- ① 왼손바닥이 위쪽으로 향하는 모션을 취하게 되면 LeapMotion UI가 화면에 나타나게 되며 UI는 동적 변화요소 UI로 PC모드와 동일하게 Event를 발생시킬 수 있으며 오른손 검지를 이용하여 선택을 할 수 있다.



- ① 왼손을 주먹을 쥔 상태에서 손을 상하좌우로 움직이게 되면 해당 방향으로 카메라의 시점, 즉 사용자 화면의 시점이 회전하게 된다. 동일하게 왼손을 주먹을 쥔 상태에서 오른손의 손가락을 이용하여 앞뒤좌우로 이동이 가능하며 이를 이용하여 더 편리하게 카메라를 이동시킬 수 있다.

6. 프로젝트 마무리

A. 기대효과

- ✓ VR과 Leap Motion을 결합해 기존 GUI 건축 설계한 틀을 깬 NUI 환경을 지원, 보다 생동감 있는 건축 설계를 보장한다. 미래에 기기들이 발전을 한다면 상상이 현실로 이루어질 수 있다.
- ✓ 일반적인 건축 시뮬레이션이 아니라 추가로 날씨변화, 재해 등의 상황을 부여해 특정 상황에서 어떻게 사람들이 행동하고 대처하는지 확인해 볼 수 있게 하는 시뮬레이션을 추가해 건축시에 참고 할 수 있다.
- ✓ 사용자가 건축 설계 시 실시간 홀로그램을 지원한 진행상황을 볼 수 있다. 기존 건축 시뮬레이션 조형도 설계에 시간을 들일 필요 없이 홀로그램을 통해서 입체감 있는 건축 시뮬레이션 조형도를 실시간 출력이 가능하다.

B. 문제점

- ✓ 자유롭게 원하는 대로 건축 시뮬레이션을 하기 위해서 프로그램내 기본적으로 제공하는 오브젝트만이 아니라 사용자가 직접 다양한 건축물을 제작할 수 있게 하려고 했지만 블록을 쌓아서 기본적인 모양만 만들 수 있게 되었다.
- ✓ 다양한 동적 변화요소를 추가 하려고 했지만, 제한된 프로젝트 기간으로 인해 몇가지만 추가 하게 되었다.
- ✓ 립모션을 메인 컨트롤러로 사용하려 했지만 기기의 한계 때문에 세밀한 조정이 불가능하여 특정 상황에서만 사용하게 하였다. 또한 VR모드에서 사용하기 위해 VR기기에 장착하려 했는데 PC모드와 VR모드의 전환이 제대로 되지 않아서 사용하지 못했다.
- ✓ AI 길 찾기 경로 구성 중 Bake 문제: 현 Unity에서 지원하는 기술 중에선 동적으로 경로를 Bake 할 수 없다는 것이 문제가 되어 동적으로 길을 생성하는 본 프로젝트에서 추가된 길을 경로로 읽기 못하는 현상이 발생
- ✓ SetDestination 버그: Unity 자체버그로 코너가 많거나 경로 계산이 안될 경우 해당 경로의 검색을 중지 캐릭터가 이동하지 못하는 상태가 발생

- ✓ Car_Object 충돌 버그: 도착지점 목적지의 폭이 좁아 자동차끼리 부딪히는 현상이 발생

C. 개선방안

- ✓ 현재 진행된 프로젝트는 크게 ON/OFF모드 두가지로 구현되어 있는데 추가적으로 자체제작모드를 추가하여 사전에 제작된 오브젝트 이외에도 사용자가 원하는 오브젝트를 직접 제작하고 이를 시뮬레이션 프로그램에서도 사용할 수 있게 한다.
- ✓ 우리가 구현한 동적변화요소는 상황변화요소 4가지, 환경변화요소 3가지이다. 추후에 사용자가 원하는 동적변화요소가 있다면 이를 구현하고 추가하여 시뮬레이션 프로그램에 적용시킬 수 있다.
- ✓ 립모션의 현재 개발상황으로는 LeapMotion 인식센서의 반대편에 있는 손모션을 정확히 인식하지 못하는 점, 인식거리의 한계, 손 모션의 떨림현상, 두 손이 겹쳤을 때 두 손 중 한 손만 인식하는 점 등을 직접 해결할 수 없으므로 손 뿐 아니라 몸 전체 모션을 인식하는 NUI기기의 변경(오쿨러스 리프트, 키넥트 등)을 통해 대체해야 한다.
- ✓ 현재 제작된 홀로그램 출력장치는 크기가 작다는 문제점을 갖고 있다. 하지만 주어진 환경안에서 최대한으로 출력한 크기이며 이를 개선하기 위해서는 큰 사이즈의 출력 모니터를 사용하여 크기에 맞게 홀로그램 출력장치를 제작한다면 더 큰 홀로그램을 출력할 수 있다.
- ✓ AI 길 찾기 경로 구성 중 Bake 문제 : NavMeshModifier의 예제 및 그 기능을 사용 하는 방법에 대해 터득 동적으로 Bake 기능을 추가
- ✓ SetDestination 버그 : navmesh.calculatepath()함수를 사용 해당 경로의 코너의 개수와 위치를 파악 코너를 향해 네비게이션을 구성 가장 가까운 코너를 목적지로 설정 해당 코너에 다가갔을 때 다음 코너를 목적지로 설정하는 것으로 SetDestination의 버그를 최소화
- ✓ Car_Object 충돌 버그 : 자동차간의 거리를 계산 퍼지로직을 사용해 서로간의 거리에 따른 속도를 조절하는 것으로 문제를 해결

D. 참고문헌 및 논문

- i. AI / Fuzzy
 - 유니티 게임 AI 프로그래밍 2/E / 레이 바레라, 아웅 시투 키야우 외 5인 저, 조경빈 역 / 에이콘 출판사 / 2016.05.27

E. 참고사이트

- i. Unity
 - 유니티최적화기법
<http://vallista.tistory.com/entry/Unity-%EA%B2%8C%EC%9E%84-%EC%B5%9C%EC%A0%81%ED%99%94-%EA%B8%B0%EB%B2%95>
 - 유니티 건물 배치
<https://www.youtube.com/channel/UCsqkgnqvehHYSXprvDSjAIA>
 - Json 저장과 불러오기
https://www.youtube.com/watch?v=cW_IXZzCU8s
 - 유니티 멀티디스플레이
<https://docs.unity3d.com/Manual/MultiDisplay.html>
 - AI / 경로 동적 Baking
<https://github.com/Unity-Technologies/NavMeshComponents>
<https://www.youtube.com/watch?v=n6ilxlcNpgM>
 - AI / 캐릭터 차량의 이동방향 조정
<https://www.youtube.com/watch?v=zSKVMXd4ob0>

ii. TrinusVR

- TrinusVR 사용법


<http://blog.naver.com/PostView.nhn?blogId=kin412&logNo=220502323956>

iii. LeapMotion

- LeapMotion 사용예제

<https://developer.leapmotion.com/unity>

F. 팀원 별 소감

팀원	소감
<p style="text-align: center;">조덕진</p> 	<p>대학시절 전공이 컴퓨터공학과였지만 학과내부사정과 개인사정으로 인하여 프로젝트 경험이 전무하였다. 그렇기에 이 과정을 수료하며 기존의 알던 기술을 다시 점검할 수 있었고 나 자신의 실력 역시 이전보다 많이 발전한 것 같다. 다양한 미니프로젝트 경험과 Final 프로젝트를 경험할 수 있어서 상당히 뜻 깊고 도움이 되었다. 그리고 프로젝트 초반에는 중요 Final프로젝트 팀의 팀장을 맡게 되었다보니 책임감과 부담감으로 힘들고 어려웠었다. 하지만 중반을 지나 마무리를 짓게 되면서 이 역할로 인하여도 '과정상의 협업'과 '마지막 결과'의 중요함을 깨달을 수 있었다.</p> <p>프로젝트 아이디어와 훈련과정 특성상 모션인식, AR/VR 등 많은 것을 처음 접하고 이용해보았다. 콘텐츠로서는 많이 접해보았었기 때문에 막연하고 개발에 거부감을 느꼈었다. 하지만 막상 배우면서 쉽지는 않았지만 상당히 재미와 보람이 있었고 프로젝트를 통해 개발 및 응용, 사용을 해보면서 조금 더 특별한 경험을 쌓고 스스로의 역량도 넓힌 것 같아 매우 의미 있었고 보람을 느꼈다.</p> <p>Final 프로젝트를 진행하며 팀 인원 수의 변경으로 인하여 작업량의 증가가 있었고 갑작스런 업무의 변화도 있었지만 이를 통해 다양한 시도 및 기능 구현을 경험할 수 있어서 좋았으며 팀원들간의 불화도 없이 순조롭게 진행되었다. 그리고 가장 다행스럽게 생각하는 부분 중에 하나로 제한된 프로젝트 기간 내에 구현하고자 목표로 했던 기능들과 프로젝트 전체를 완료할 수 있었다는 점이다. 팀원들이 주어진 역할을 잘해주었으며 서로 협력하였기 때문에 팀원들에게도 상당히 고마움을 느낀다.</p>

<p>김영서</p> 	<p>프로그래밍 입문자로 2개월정도 되는 시간동안 프로젝트를 진행하게 되어 걱정도 많았지만 생각보다 개발하는 과정은 재미있었다. 누군가가 가르쳐주는 것이 아닌 스스로 알아가야 했다. 그런 과정들이 당시에는 매우 힘들었지만 혼자서 해결하지 못한 일들을 조원들과 같이 생각하고 해결하면서 단순히 지식만 얻은 것이 아니라 문제를 헤쳐 나가는 인내심과 자신감을 얻었다. 그리고 프로그래밍을 할 때, 정말 사소한 부분인데, 경험이 부족하여 어떤 것을 추가해야 하는지 모르고 또는 주어진 정보가 너무 적어서 고생을 한 경우가 많았는데 문제들을 고민하고 해결해 나가면서 문제 해결할 수 있는 능력을 갖게 된 것 같다.</p>
<p>이태경</p> 	<p>수 많은 프로젝트를 하며 성공의 달콤함과 실패의 쓴맛을 반복하며 온 이번 프로젝트는 첫 장기 프로젝트라는 기대감과 실패의 불안감에 열성적으로 한 프로젝트였습니다. 열성적인 덕에 보다 좋은 프로젝트 결과를 내기 위해 나갔지만 그 때문에 팀원과의 불화와 소통의 부재로 어려움을 겪으며 자신의 주장 보단 다른 사람들의 주장을 먼저 존중해야 하는 자세가 필요하다는 것을 느꼈으며 제 자신을 바꾸기로 마음 먹었습니다. 그래도 저의 고치려고 노력한 성격과 함께 팀원들과 소통하고 화합을 맞춰 성공적인 프로젝트가 되어 모두가 만족하는 모습에 한편으로는 안도감과 프로젝트에 대한 자심감을 얻어 가장 많은 것을 느낀 것과 동시에 가장 뜻 깊은 프로젝트였습니다. 마지막으로 길고 길었던 프로젝트를 마치며 여기까지 버텨와 준 팀원들과 팀장님께 감사의 말씀을 드리고자 합니다.</p>