한국직업능력교육원(시흥)

VR 비계 전도 체험

김성수

목차

01 프로젝트 개요

▪ 프로젝트 주제

공사 현장의 비계 전도 추락 사고를 체험해볼 수 있다.

▪ 프로젝트 개요

공사 현장 안전 사고를 가상 공간에서 체험해 봄으로써 작업자에게 위험성을 인지 시키고 안전장구의 사용을 독려할 수 있다.

실무 프로젝트에서 요구되는 3D 모델링 및 Unity VR구현 프로젝트를 진행 함으로써 포트폴리오 및 취업지원 등에 활용하도록 한다.

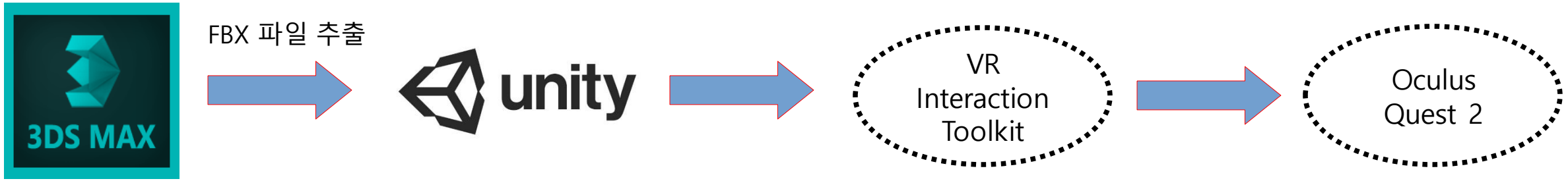
▪ 활용 장비

H/W : Dell Inspiron 11 Gen i7 2.50Hz, Oculus Quest2

S/W : Unity(2021.3.0f1 LTS), 3DS Max 2022

01 프로젝트 개요

프로젝트 제작 방식



모델링 에셋 제작

Unity 프로젝트
에셋 추가

Unity 제공 VR SDK를 활용,
Meta의 Oculus Quest 2 기기를
기반으로 구동되도록 제작

02 프로젝트 수행 절차 및 방법

1. 프로젝트 기획

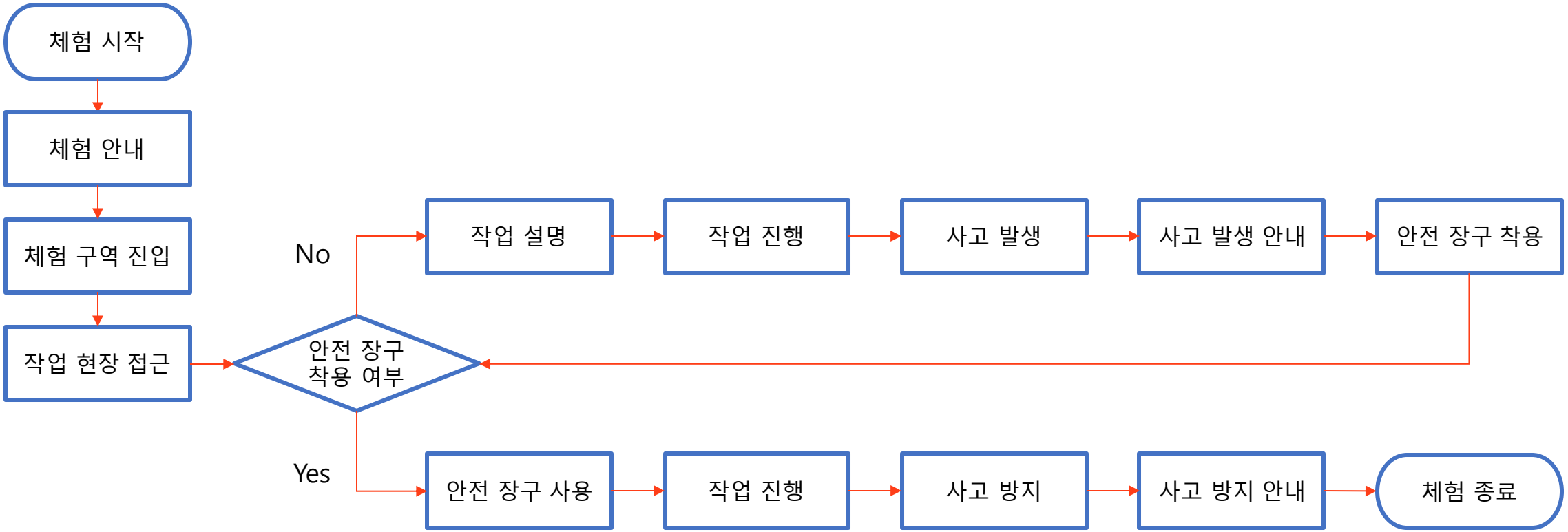
▶ 프로젝트 타임라인

구분	작업 내용	Apr																												May								
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6	7	8	9
기획	프로젝트 기획																																							
알파 버전	알파 버전 기능 구현																																							
아트 에셋	공사 현장 벽체																																							
	대형 구조물																																							
	인터렉션 오브젝트																																							
베타 버전	버그 수정 및 기능 보완																																							
	UI 이미지 추가																																							
	오디오 추가																																							
프로젝트 정리	보고서 작성																																							

02 프로젝트 수행 절차 및 방법

1. 프로젝트 기획

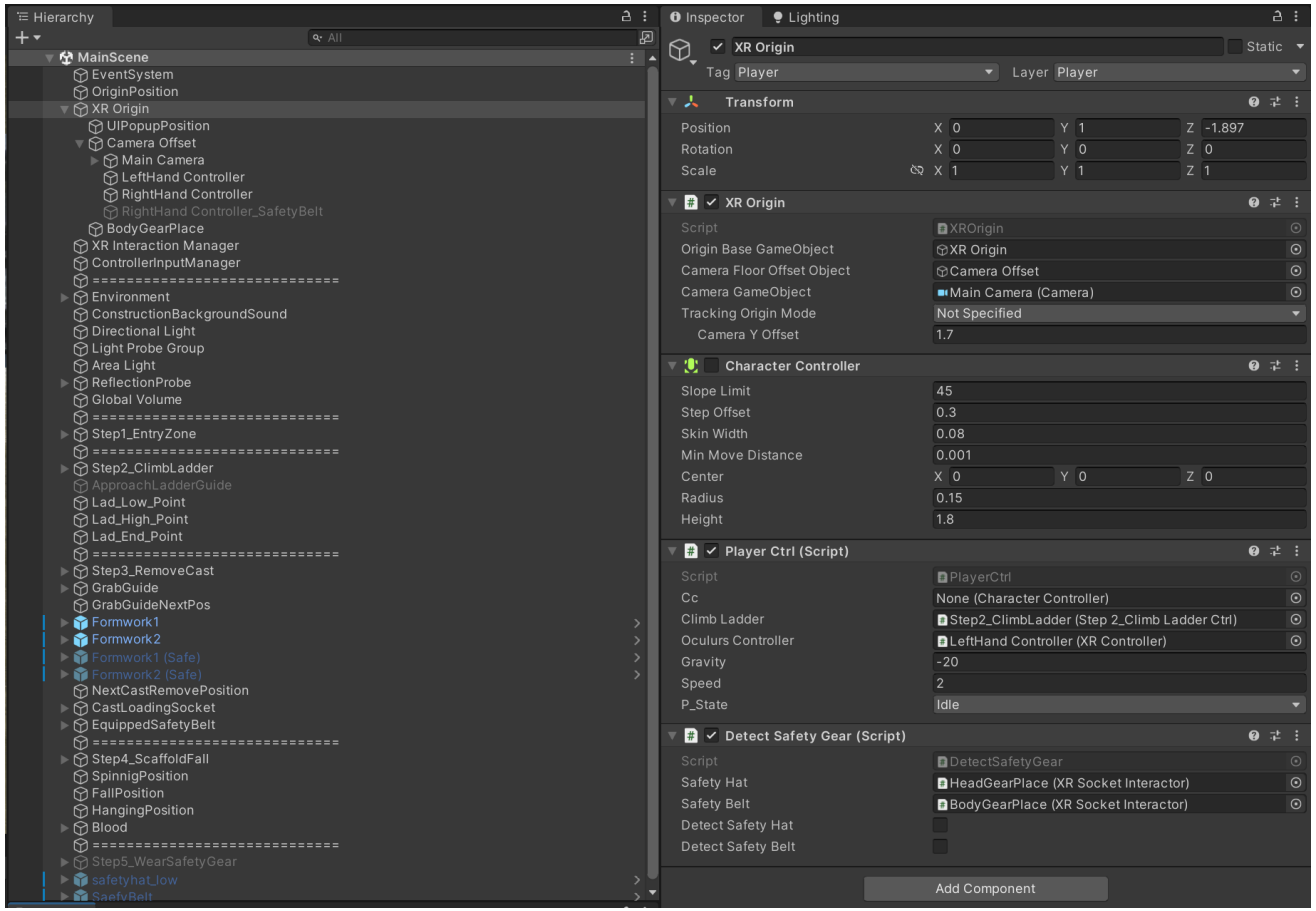
▶ 프로젝트 시나리오



02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶VR Interaction Toolkit 세팅



세팅 순서 및 내용

1. Package manager에서 Interaction Toolkit 다운로드 후 프로젝트에 추가
2. XR origin 오브젝트 추가
3. 시작 시 카메라 위치값 조정
4. Main Camera 자식 개체로 등록된 Right/left Hand Controller에 모델 프리팹 등록

   등록한 프리팹은 HandControllerPresence 스크립트를 통해 입력된 VR H/W 별 컨트롤러 모델을 출력하게 됨

02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶VR Interaction Toolkit 세팅

```
참조 1개
IEnumerator CreateInputDeviceController(float delayTime)
{
    // link 에서 시작 시 HMD만 입력을 받는 상태이기 때문에 컨트롤러 값이 들어가지 않음, 컨트롤러 입력 들어오는 시간 대기 후 실행
    yield return new WaitForSeconds(delayTime);

    // 연결되어 입력을 받는 모든 컨트롤러를 저장할 리스트
    List<InputDevice> devices = new List<InputDevice>();

    // 인풋디바이스 중 컨트롤러 특성을 가지는 디바이스만 리스트에 저장
    InputDevices.GetDevicesWithCharacteristics(controllerCharacteristics, devices);

    // HandController에 프리팹 연결 시 입력 디바이스의 이름 및 정보를 확인하기 위해 아래 작업 실행
    foreach (var item in devices)
    {
        Debug.Log(item.name + item.characteristics);
    }

    if (devices.Count > 0)
    {
        // 프리팹 생성을 위해 입력된 기기의 컨트롤러를 저장
        targetDevice = devices[0];

        // 기기별 컨트롤러 프리팹 리스트 중 현재 입력된 컨트롤러와 이름이 같은 프리팹을 찾아 저장
        // GameObject.Find의 매개변수는 람다식을 이용해 넣어줌
        // 프리팹 리스트 중 입력 컨트롤러와 이름이 같은 것을 controller 변수로 저장 후 찾기
        GameObject ctrlPrefab = controllerPrefabs.Find(controller => controller.name == targetDevice.name);

        // ctrlPrefab가 저장되면 Instantiate한다
        if (ctrlPrefab)
        {
            spawnCtrl = Instantiate(ctrlPrefab, this.transform);
        }
        // 입력된 컨트롤러가 리스트에 없을 시 에러 문구 출력 후 디폴트 프리팹 생성
        else
        {
            Debug.LogError("연결된 VR 기기의 컨트롤러 모델을 찾지 못 했습니다.");
            spawnCtrl = Instantiate(controllerPrefabs[0], this.transform);
        }

        spawnHandModel = Instantiate(handPrefab, this.transform);
        handAnim = spawnHandModel.GetComponent<Animator>();
    }
}
```

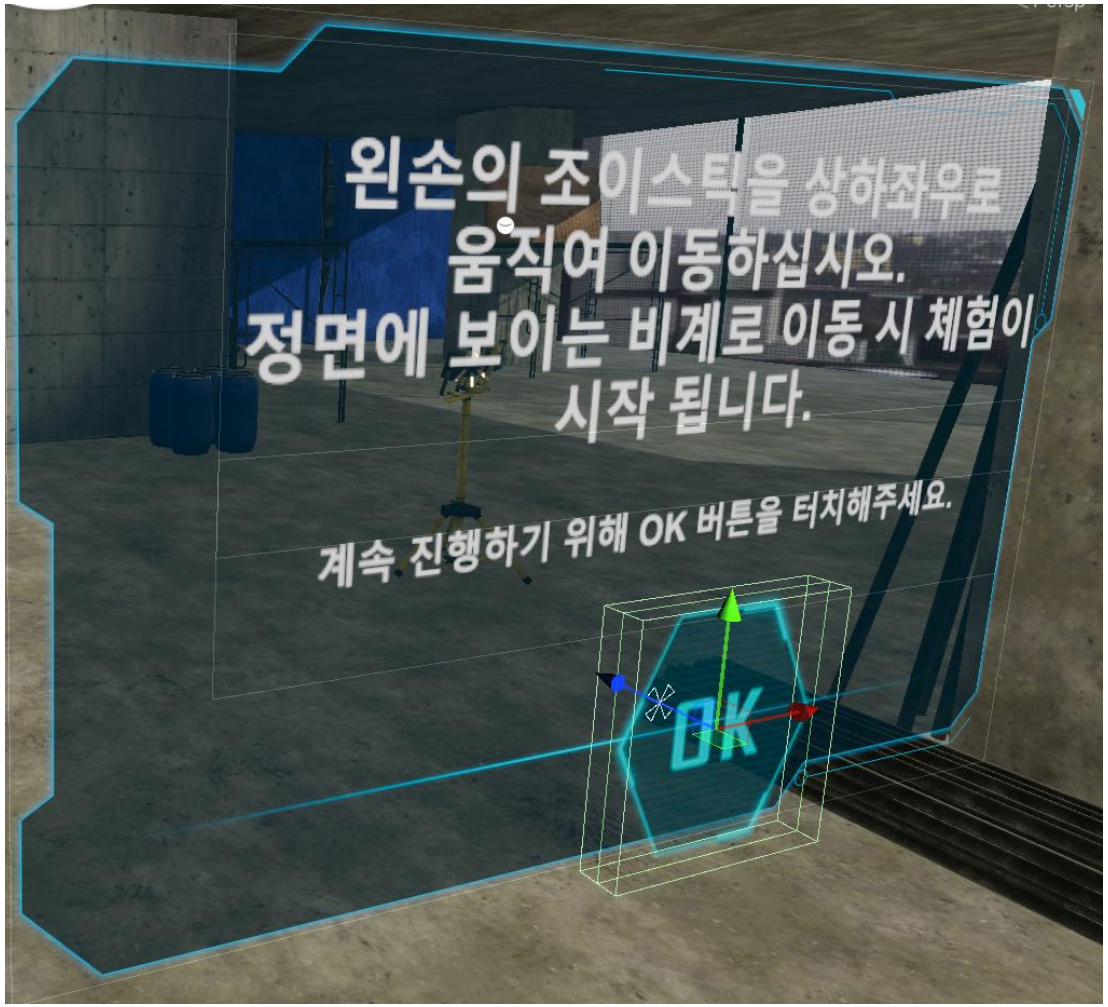
HandControllerPresence

1. VR H/W 컨트롤러 입력 대기
2. 입력된 컨트롤러 정보를 저장할 device 리스트 선언
3. 컨트롤러 특성을 가지는 H/W 만 리스트에 저장
4. 입력된 컨트롤러의 이름 및 정보를 콘솔창에 출력
5. HandController에 입력된 컨트롤러와 이름이 같은 프리팹 생성
    1) public List 로 선언된 controllerPrefabs에 들어가 있는 프리팹 중 입력 컨트롤러와 이름이 같은 것을 ctrlPrefab 로 저장
    2) HandController 위치에 생성
    3) 이름이 같은 프리팹이 없을 경우 기본 모델을 HandController 위치에 생성 후 에러 문구 출력

02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶안내 UI 및 단계별 진행



체험 존 collider에 trigger enter 시 체험이 시작 되며

각 안내 UI는 OK 버튼과 hand controller 의 collider trigger enter 시 다음 단계로 진행 되는 방식으로 구성됨

```
Unity 메시지 | 참조 0개
private void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("ControllerLeft") ||
        other.CompareTag("ControllerRight"))
    {
        audioManager.PlayAudioOnce(select);

        player.GetComponent<CharacterController>().enabled = true;

        Destroy(this.gameObject);
    }
}
```

02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶거푸집 잡기 및 제거

```
// Interactable의 업데이트 진행
참조 0개
public override void ProcessInteractable(XRInteractionUpdateOrder.UpdatePhase updatePhase)
{
    if(secondHandInteractor && selectingInteractor)
    {
        /*/ 참고 자료 원본 스크립트(z 축으로 길게 잡을 때 적용)
        selectingInteractor.attachTransform.rotation
            = Quaternion.LookRotation
        (secondHandInteractor.attachTransform.position - selectingInteractor.attachTransform.position);
        //*/

        // 수정본(x 축으로 길게 잡을 때 사용)
        selectingInteractor.attachTransform.LookAt
            (secondHandInteractor.attachTransform.position, selectingInteractor.attachTransform.right);

        selectingInteractor.attachTransform.RotateAround
            (selectingInteractor.attachTransform.position, selectingInteractor.attachTransform.right, 90);

        selectingInteractor.attachTransform.RotateAround
            (selectingInteractor.attachTransform.position, selectingInteractor.attachTransform.up, 90);
    }
    base.ProcessInteractable(updatePhase);
}
```

RemovePlateInteractable 스크립트

동일 오브젝트에 두 개의 그랩 인터랙터가 들어올 수 있도록 설정 후 첫번째 인터랙터가 두번째 인터랙터와 상호작용할 수 있도록함

1. ProcessInteractable을 override 해 업데이트 동안 해당 상호작용 내용이 실행되도록 함
2. 첫번째 두번째 Interactor의 값이 들어왔을 때 상호작용
3. 첫번째 Interactor의 z축이 두번째 Interactor를 계속 바라보도록 설정하여 두번째 Interactor의 움직임을 따라 오브젝트가 회전함
4. 해당 잡기 동작에선 손등이 위로 가게 잡기 때문에 두번째 Interactor를 바라보는 축은 y축이 되어야 함
5. 첫번째 Interactor 위치를 기준으로 회전 시켜 y축이 두번째 Interactor를 보도록 수정

02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶전도 추락 애니메이션

```
// 추락 인명 사고 애니메이션
참조 1개
void FallBehind()
{
    player.transform.DOMove(spinningPos.position, 2.0f);
    player.transform.DOShakePosition(1.5f, axisShakeStrength, vibrato, randomness).SetDelay(0.5f);
    player.transform.DORotateQuaternion(spinningPos.rotation, 1.5f).SetDelay(1.8f);
    StartCoroutine(audioManager.PlayAudioDelayed(fallDie, 2.8f));
    player.transform.DOMove(fallPos.position, 1.5f).SetEase(Ease.OutBounce).SetDelay(3.0f);
    player.transform.DORotateQuaternion(fallPos.rotation, 0.5f).SetDelay(3.5f);
    blood.DOScale(1.0f, 0.5f).SetDelay(3.5f);
}

// 추락 인명 사고 방지 애니메이션
참조 1개
void FallSafely()
{
    player.transform.DOMove(spinningPos.position, 2.0f);
    player.transform.DOShakePosition(1.5f, axisShakeStrength, vibrato, randomness).SetDelay(0.5f);
    player.transform.DORotateQuaternion(spinningPos.rotation, 1.5f).SetDelay(1.8f);
    StartCoroutine(audioManager.PlayAudioDelayed(fallDie, 3.0f));
    player.transform.DOMove(hangingPos.position, 1.0f).SetDelay(3.0f);
    player.transform.DORotateQuaternion(hangingPos.rotation, 0.5f).SetDelay(3.5f);
    player.transform.DOShakePosition(7.0f, 0.4f, 2, 0.0f).SetDelay(3.9f);
}
```

DOtween 애니메이션

1. 이동과 회전의 기준점이 되는 empty object 생성
2. 배치 후 해당 오브젝트의 포지션, 로테이션 값을 따라 움직이도록 설정
3. SetDelay 함수로 각 움직임이 순차적으로 진행되도록 설정
4. SetEase 함수로 추락 후 바운스, 추락 중 줄에 걸리는 디테일 설정

02 프로젝트 수행 절차 및 방법

2. 알파 버전 기능 구현

▶안전장구 착용

(에디터 화면)

(사용자 화면)

Socket Interactable

1. 안전장구 프리팹에 Grab Interactable 컴포넌트 추가
   머리와 몸체 착용될 부분에 Socket Interactor 추가

2. Socket Interactor 는 해당 범위에 들어간 Interactable 오브젝트를 자신의 어태치 포인트로 재위치 시킴

3. 안전 장구 착용 프로세스

   1) 컨트롤러로 잡는다.
   (Grab Interactor 어태치 포인트로 재위치)

   2) 머리 부분에 갖다 댄다.
   (Socket Interactor triggerenter 인식)

   3) 손을 놓는다.
   (Socket Interactor 어태치 포인트로 재위치)

02 프로젝트 수행 절차 및 방법

3. 3D 모델링 에셋 추가

▶사용 3D 모델 리스트

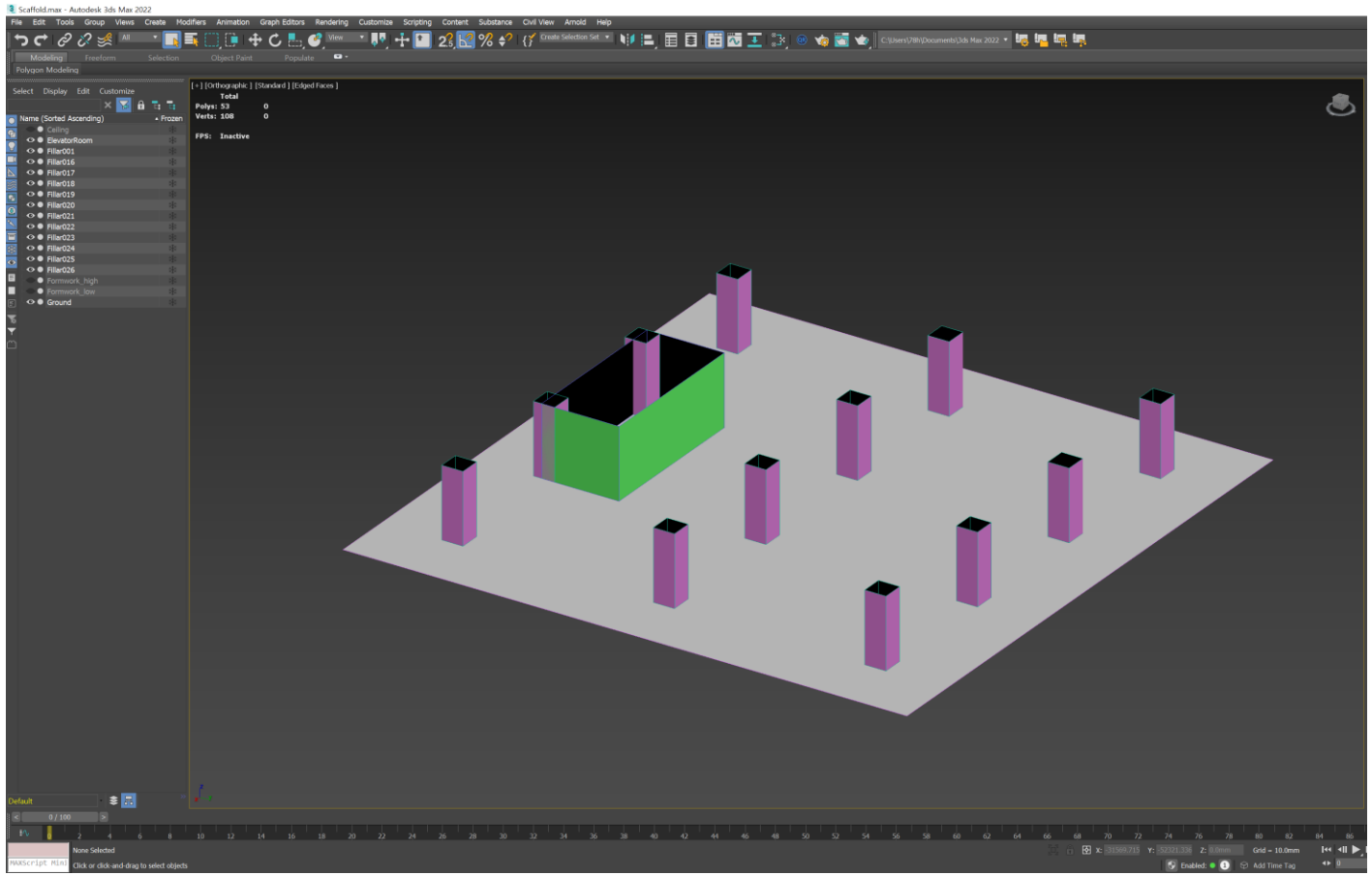
구분	아트 에셋	제작 방법
공사 현장 벽체	바닥	3ds Max 이용 제작
	천장	
	기둥	
대형 프랍	거푸집 더미	
	파이프 더미	
	방진, 방음 천막	Unity asset store 다운로드
	비계	
소형 프랍	스포트라이트	URP Sample asset 활용
	공구 상자	
	작업대	
	드럼통	Free resource
	파이프	3ds Max 이용 제작
인터렉션 오브젝트	거푸집	
HDRI	HDRI	Free resource
UI 이미지	UI 배경	
	OK 버튼	
오디오	음성 안내	Papago 음성 서비스
	공사 현장 배경음	Free resource
	버튼 선택음	
	거푸집 제거 및 적재	
	비명 소리	
	안전장구 착용	

1. 벽체 및 주요 인터렉션 오브젝트를 기획 의도에 맞게 자체 제작
2. 비계, 천막, 드럼통 등 일반적인 규격이 있는 프랍은 기존에 제작된 에셋을 최대한 활용
3. UI 이미지는 협력 업체 제공 에셋 활용
4. 오디오는 음향 무료 에셋 사이트 자료를 사용했으며, 음성 안내의 경우 Naver Papago의 음성 서비스를 녹음해 사용

02 프로젝트 수행 절차 및 방법

3. 3D 모델링 에셋 추가

▶공사 현장 벽체



3ds Max를 이용 벽체 제작

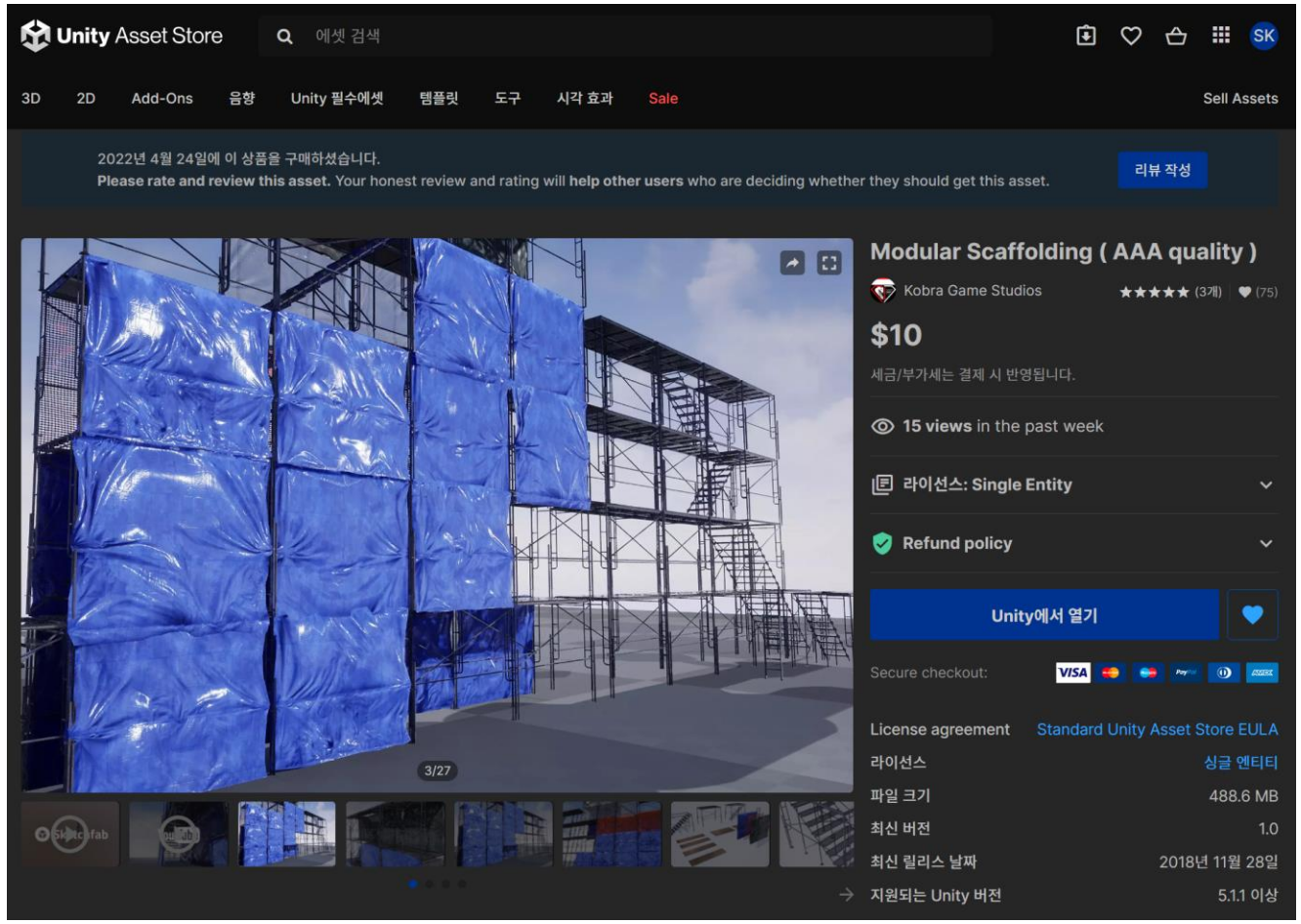
1. 고층 빌딩 공사 현장 중 한 층에 대한 씬으로 설정 후 제작

2. 최적화를 위해 화면 상에 보이지 않는 부분인 위, 아래 면은 삭제

02 프로젝트 수행 절차 및 방법

3. 3D 모델링 에셋 추가

▶ **대형 구조물**



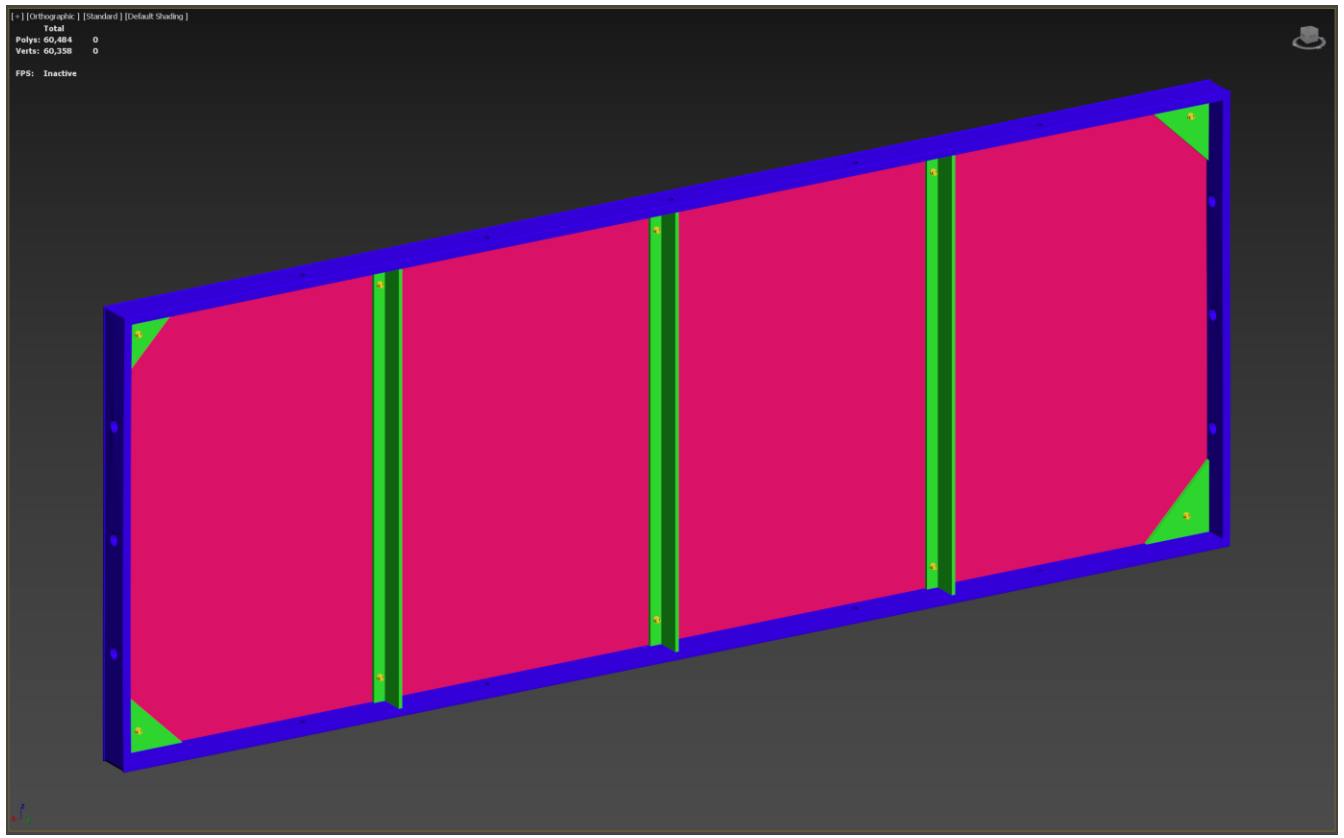
비계, 천막 및 작업대 등

1. URP 샘플 에셋 및 에셋 스토어 에셋 이용
2. LOD가 들어간 에셋들로서 최적화에 용이

02 프로젝트 수행 절차 및 방법

3. 3D 모델링 에셋 추가

▶인터렉션 오브젝트



유로폼 거푸집

1. 3ds Max를 통해 하이폴리곤 모델 제작
2. 리토폴로지 작업을 통해 버텍스 수를 60,000개 → 1,500개로 줄임
3. 노말맵 추출을 통해 표현의 디테일은 최대한 살림
4. 상호작용 시 Unity 좌표계에 맞도록 y-axis up 방향으로 수정

02 프로젝트 수행 절차 및 방법

4. 베타 버전

▶버그 수정 및 기능 보완

```
public void StartSafetyBeltFix()
{
    // 안전대 결착 안내 UI와 음성 출력
    fixInstructionUI.SetActive(true);
    audioManager.PlayAudioOnce(fixSafetyGear);

    // 안전대 결착 위치 가이드 출력 및
    // 기존 컨트롤러 비활성 후 안전대 결착 컨트롤러 활성
    safetyBeltGuide.SetActive(true);
    originController.SetActive(false);
    safetyBeltController.SetActive(true);
}

Unity 메시지 | 참조 0개
private void OnTriggerEnter(Collider other)
{
    StartCoroutine(SafetyBeltFixed(other));
}

참조 1개
IEnumerator SafetyBeltFixed(Collider collider)
{
    // 오른손 컨트롤러 감지 시 안전대 결착 완료 프로세스 실행
    if (collider.CompareTag("ControllerRight"))
    {
        // 안전대 결착 안내 UI 비활성 후 결착 사운드 출력
        fixInstructionUI.SetActive(false);
        audioManager.PlayAudioOnce(fixedSound);

        // 컨트롤러 모델 원복 후 안전대 모델 출력
        safetyBeltController.SetActive(false);
        originController.SetActive(true);
        safetyBeltGuide.SetActive(false);
        safetyBelt.SetActive(true);

        yield return new WaitForSeconds(1.5f);

        // 거푸집 제거 프로세스 시작
        RC.StartInstruction();
    }
}
```

안전대 결착 프로세스 추가

1. 안전장구 착용 완료 시 플레이어의 PlayerState를 Safety 상태로 전환
2. 안전 장구 착용 후 작업 현장 접근 단계에서 플레이어의 PlayerState 체크 후 Safety 일 경우 안전대 결착 프로세스 진행
3. 안전대 결착 위치 안내 UI 및 음성 출력 후 오른손 컨트롤러 모델을 안전대 모델로 변경
4. 결착 위치와 컨트롤러의 TriggerEnter 발생 시 결착 완료 프로세스 진행
5. 안내 UI 비활성 및 결착 사운드 출력, 컨트롤러 모델 원복 진행
6. 거푸집 제거 작업 안내 프로세스 시작

02 프로젝트 수행 절차 및 방법

4. 베타 버전

▶UI 이미지 에셋 추가

```
SubShader
{
    Tags
    {
        "Queue"="Transparent"
        "IgnoreProjector"="True"
        "RenderType"="Transparent"
        "PreviewType"="Plane"
        "CanUseSpriteAtlas"="True"
    }

    Stencil
    {
        Ref [_Stencil]
        Comp [_StencilComp]
        Pass [_StencilOp]
        ReadMask [_StencilReadMask]
        WriteMask [_StencilWriteMask]
    }

    Cull Off
    Lighting Off
    ZWrite Off
    ZTest Always
    Blend One OneMinusSrcAlpha
    ColorMask [_ColorMask]

    Pass
    {
        Name "Default"
    CGPROGRAM
        #pragma vertex vert
        #pragma fragment frag
        #pragma target 2.0
```

UI 이미지 에셋

1. 제공된 이미지를 사용 임시 이미지 대체

2. UI 셰이더 옵션 수정을 통해 우선 순위로 렌더링 되도록 수정하여 다른 오브젝트에 의해 가려지지 않도록함

02 프로젝트 수행 절차 및 방법

4. 베타 버전

▶오디오 추가

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

Unity 스크립트(자산 참조 1개)|참조 6개
public class AudioManager : MonoBehaviour
{
    public AudioSource source;

    참조 20개
    public void PlayAudioOnce(AudioClip clip)
    {
        source.clip = clip;
        source.Play();
    }

    참조 2개
    public IEnumerator PlayAudioDelayed(AudioClip clip, float time)
    {
        source.clip = clip;

        yield return new WaitForSeconds(time);

        source.Play();
    }
}
```

오디오

1. 오디오 무료 공유 사이트, 파파고 음성 서비스 이용 안내 음성 에셋을 사용

2. 별도의 오디오 매니저를 만들어 하나의 오디오 소스를 통해 재생될 수 있도록 관리

   사용 시 각 메소드를 호출해 사용

3. PlayAudioOnce

   1) 들어온 clip 파라미터를 해당 AudioSource의 clip으로 설정 후 1회 플레이

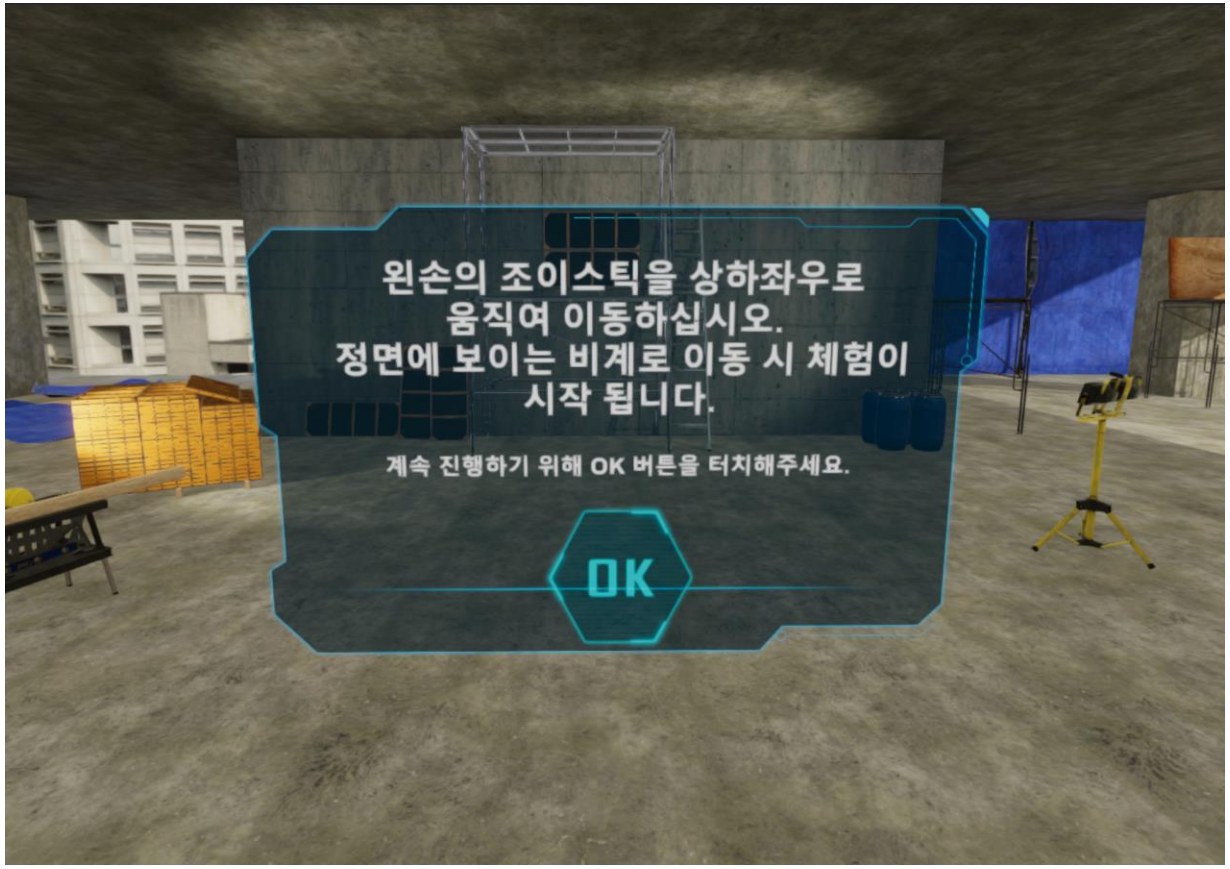
4. PlayAudioDelayed

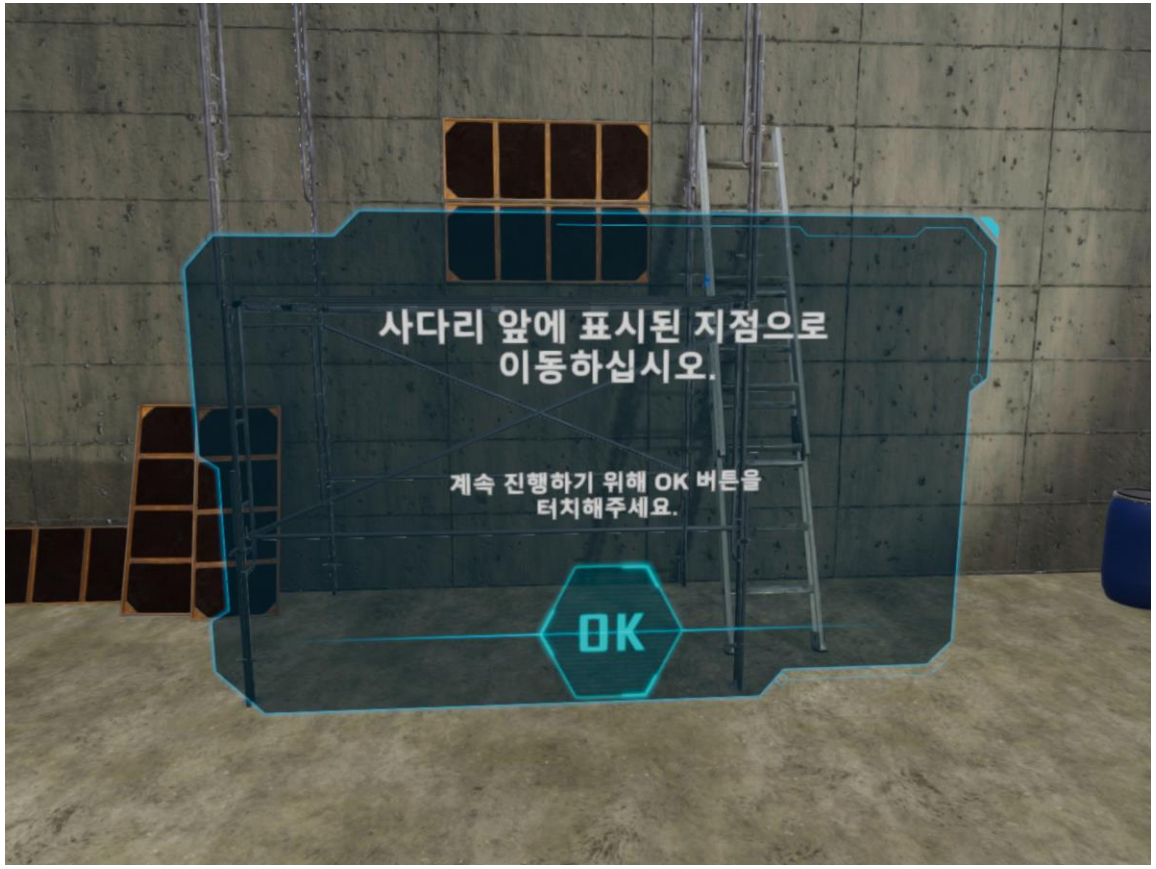
   1) 들어온 clip 파라미터를 해당 AudioSource의 clip으로 설정

   2) time 파라미터의 시간 만큼 지연 후 플레이

03 프로젝트 수행 결과

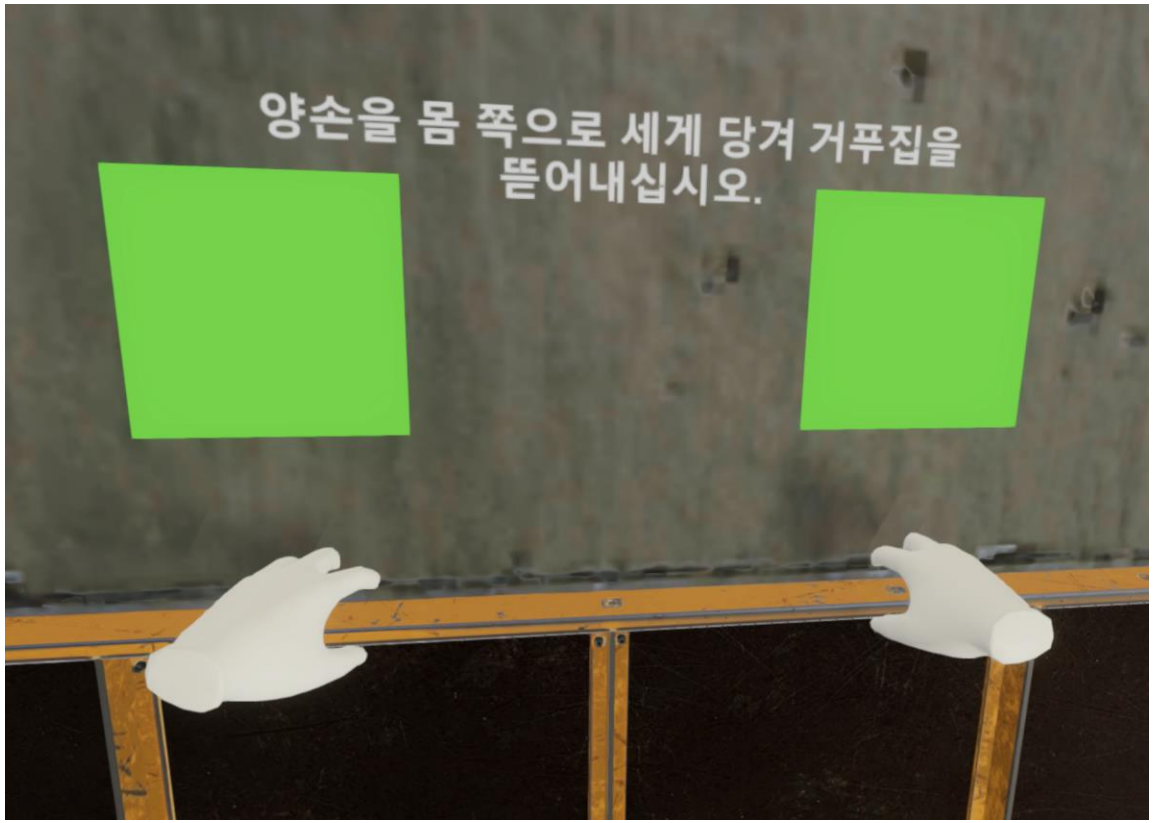
1. 체험 시작 안내





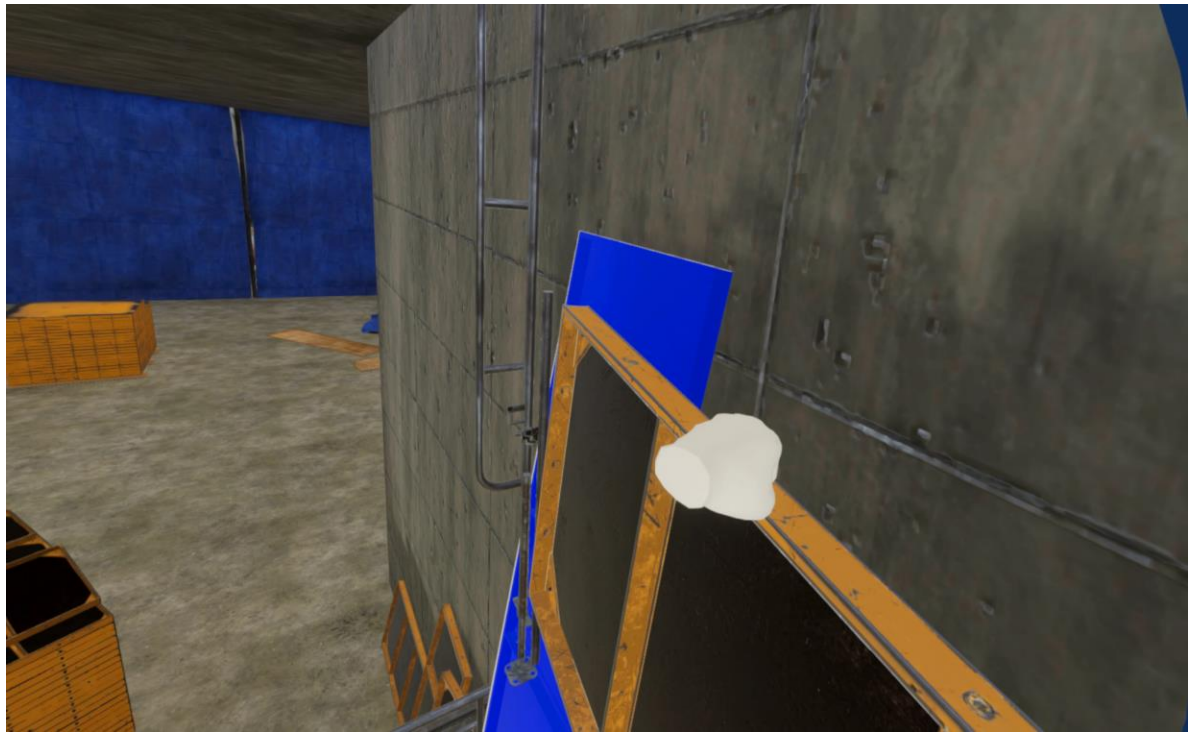
03 프로젝트 수행 결과

2. 거푸집 제거 작업 안내



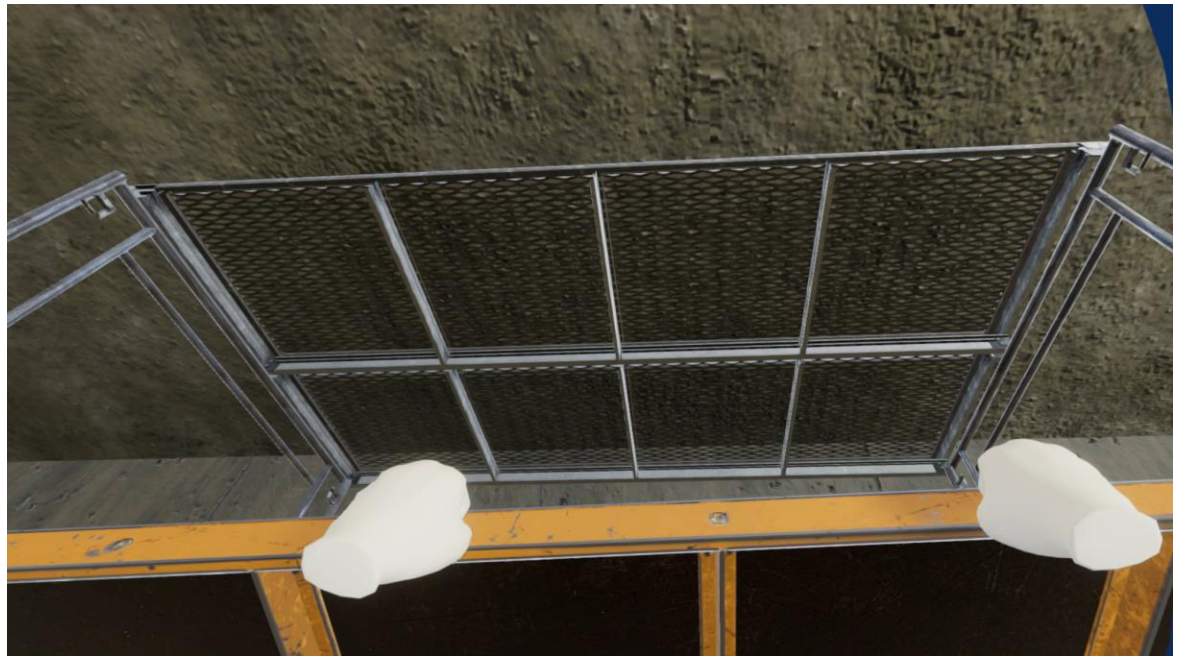
03 프로젝트 수행 결과

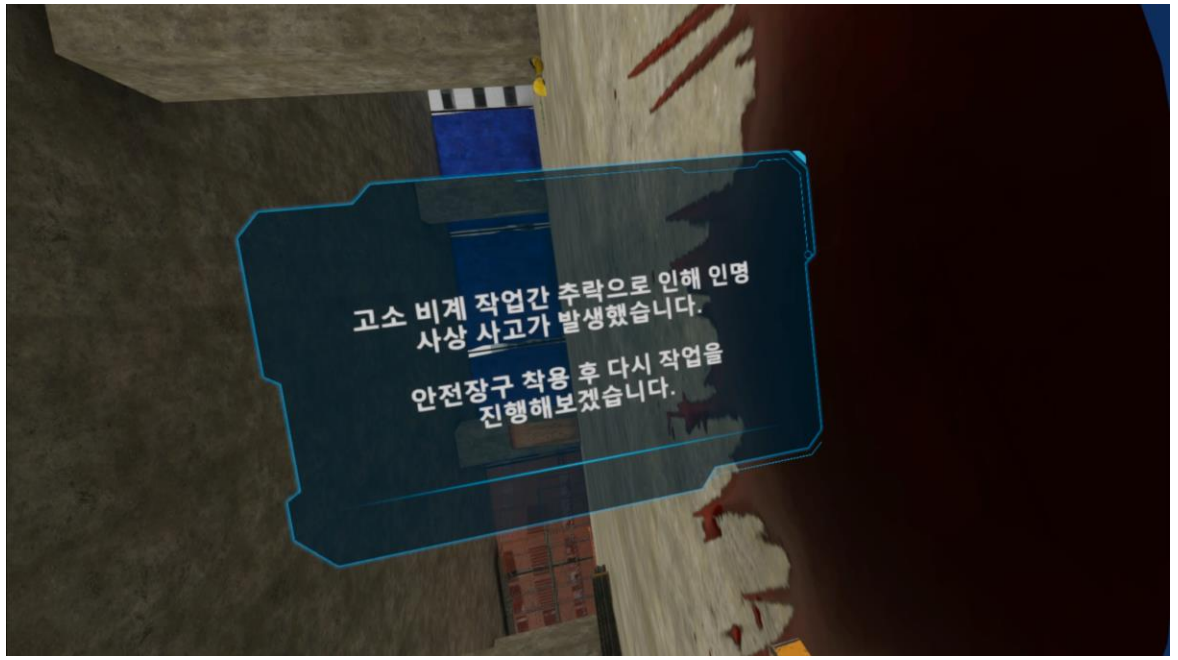
3. 거푸집 제거



03 프로젝트 수행 결과

4. 추락 사고 발생





03 프로젝트 수행 결과

5. 안전 장비 착용

03 프로젝트 수행 결과

6. 사고 방지