2023

Hoon Seok Kim

https://github.com/kkhese/VRproject

CS 458 Project Report

VIRTUAL AND AUGMENTED REALITY

Games for World Kimchi Canada Festival & Expo 3
1. Introduction/Background 3
2. Overall Plan and Features 3
3. Design Process and Comments 6
1. Menu: Background video, Main screen, Scene manager – Scene: SelectS 6
1.1 360 Degree Panoramic Video Play for Background 6
1.2 Main Screen: Half transparent screen video play 7
1.3 Scene Manager 8
2. Mini-Game: K-pop Guardian 14
2.1 Background: Skybox, Terrain, Objects 14
2.2 Effects: Crosshair, Bullet, Stick 16
2.3 Add enemies 17
2.4 Add additional characters 20
2.5 UI 20
2.6 Attack and Game Over 20
3. Exhibition Halls: 6band & Viviz – Scene Level3 & Level4 21
3.1. 6band: Stress resolving performance 21
3.2. Viviz: Exhibition Hall with Ecstasy 27
4. Functions: User Manual 29
1. How to start 29
2. Exhibition Hall 29
2.1. ViViZ Hall 29
2.2. 6band Hall 30
3. Mini Game: K-pop Guardian 30
3.1. Characters 30
3.2. Weapons 31
5. Future Work 32
1. Short Term Work 32
1.1 K-pop Guardian 32
1.2 6band Exhibition Hall 32
1.3 ViViZ hall 32
2. Long Term Work 32
1.1 More Mini Games 32
1.2 More Exhibition Halls for 3 other artists 32
1.3 Exhibition Halls for June 25th 32

Games for World Kimchi Canada Festival & Expo

1. Introduction/Background

As VR is getting more and more popular, everyone is thinking about using VR for promotions or advertisements. However, it is not effective because VR devices are not widely distributed to the general public like TV or mobile phones. That is why big companies like Adidas and McDonald's are only doing it on a trial basis to instill in the public an image that they are leading the new era [1,2]. Therefore, promoting in VR for a specific event has huge doubts in effectiveness. However, it is true that people are interested in VR when we see people visiting VR rooms and enjoying VR experiences in their spare time.

The VR application to be created through this project is for promotion of the event, but it is not intended to promote participation in the event. It would be effective to rely on TV or mobile phones as a medium to promote participation in the event. The purpose of this VR application is to enable people who visit on the day of the event to receive information related to the event in a fun way at the entrance of the event hall before participating in the event in earnest. Speaking of food, it becomes an appetizer before the main dish.

The coming event, festival & expo, which is expected to be the largest not only in Canada but also worldwide among all Korean cultural-themed events, will be held in Regina this year, 2023. The two-day festival and expo, which will be held on June 24 and 25 at Mosaic Stadium, will feature taekwondo, hanbok, and K-pop concerts, led by Korean food culture Kimchi [3].

To be developed application is divided into two main parts. There is an exhibition hall part that introduces the artists participating in the night music festival, and a mini game part based on the contents related to the event. Since this is a project using the Unity engine, a game engine, game elements were implemented as many as possible.

The supported VR equipment is **META OCULUS 2**.

2. Overall Plan and Features

The coming event, festival & expo, which is expected to be the largest not only in Canada but also worldwide among all Korean cultural-themed events, will be held in Regina this year, 2023. The two-day festival and expo, which will be held on June 24 and 25 at Mosaic Stadium, will feature taekwondo, hanbok, and K-pop concerts, led by Korean food culture Kimchi [3]. This course project is only for the event on June 24th. The application starts from the menu with the opening video. The mini-game part will consist of a total of four, and only one has yet been released in this project. A total of 10 K-pop artists will be invited, 5 on the 24th and 5 on the 25th. Of the five exhibition halls for singers invited on the 24th, only two have been built till now. The opening video for the 25th is scheduled to be completed in May. The mini game part will not change, only the artists' exhibition hall will change.

0.Version: Beta 0.5(Figure 1 shows artist line up on 24th June)

Included

Menu, Mini game: K-pop Guardian, Exhibition hall: 6band and Viviz on 24th June.

To be updated by the first week of May

Mini game: Taekwondo, Hanbok, Kimchi, Exhibition hall: CheeTah, SunYe, OneUs on 24th June.

To be updated by the second week of June

Menu, Exhibition hall for 6 more artists(to be announced on May) on 25th June.

Figure 1 Artist line up on 24th June

1.Menu (Done at this project)

Since this game has the nature of promoting a specific event, and the highlight of this event is K-pop night music festival. The opening of the game starts with a video showing the introduction of invited K-pop stars. Not 3D but **panoramic surrounded screen** was used for playing video. The similar technique exist in YouTube service. YouTube also offers 360 degree VR video service. It makes user turn around all directions to check the whole video screen so it is very immersive. Video is about 1 min and it ends with Mosaic Stadium image. If it is longer than 2 mins, users tend to feel boring. **Menu icons** stay at the front. There is a flat **half transparent sub-screen** at the front, which shows promoting footage for the event and it ends up with showing event information. **Dynamic text** exists on the top of screen. The displaying of text ends up with the title of the event.

2.Mini-Game(K-pop Guardian)

Figure 2 Tower defense

This game was inspired by general **tower defense** game shown in figure 2 [5], but the purpose of the game is to **secure the gate** entrance until the K-pop star arrives. It is a game that has been reproduced so that anyone can easily experience the situation as a game in preparation for a large crowd when a singer appears on the day of the event. Many fans rush to the gate and some unwelcomed celebrities are trying to come into the stadium. The player needs to push back or destroy incoming people or celebrities, so that the star can come into the stadium safely. Three different enemies exist, and the game will be over if the player's HP becomes 0. If a star comes into the stadium. The game will be successfully over.

3.Exhibition hall for 6Band

In this exhibition hall, 6band's representative hit song, ‘I Want to Eat Banana’ is played. Because the song is heavy metal, stress resolving elements were added. When the player hits the word ‘BANANA’ or banana object with sticks, it reacted. It made players swing the controller wildly with loud music.

4.Exhibition hall for Viviz

A completely black background was used so that the girl group's appearance could shine the brightest. There is a structure created by stacking 20 spheres. Outside light was blocked in the confined space so that you can fully immerse yourself in the visual of Viviz. Eight different colors of light were shot in all directions from the inside, creating a sense of mystery. The inside of the structure has four tiers, and each tier simultaneously screens a different music video to create a voluminous surround music video hall. As for music, playing 4 songs at the same time makes people dizzy, so only the most popular representative song, 'Pull Up' is played.

4.Missing contents 1 – mini games

4-1. Taekwondo

A game will be similar to fruit ninja, which will be designed to break pine boards instead of fruits. It's better to be developed to kick, but there is a constraint of holding a controller in hands.

4-2. Han-bok Decoration

It will be designed so that users can wear hanbok on their avatars and easily apply make-up as a common avatar decoration.

4-2. Kimchi Master

It will be a game where people can experience making Kimchi with given ingredients. The players can make various kinds of Kimchi according to their order. This game could be described as a much simplified version of "Over Cooked".

5.Missing contents 2 – Exhibition halls for Cheetah, Sunye, Oneus

A novel idea is required to make the stars stand out while maximizing the effect of VR. Currently, the boy band exhibition hall is being considered to be decorated with beat drummers, similar to Beat Saber. It contains a mini-game element of drumming to the rhythm of their hit song, and puts the singer's face on the surface of the drum so that the singer and the hit song can be matched and memorized to the players.

3. Design Process and Comments

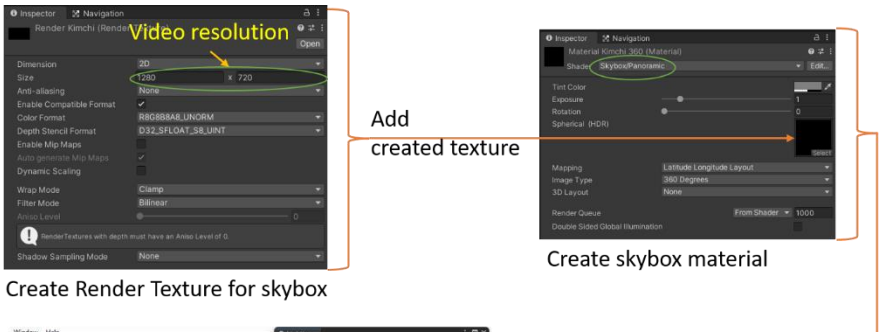
1. Menu: Background video, Main screen, Scene manager – Scene: SelectS

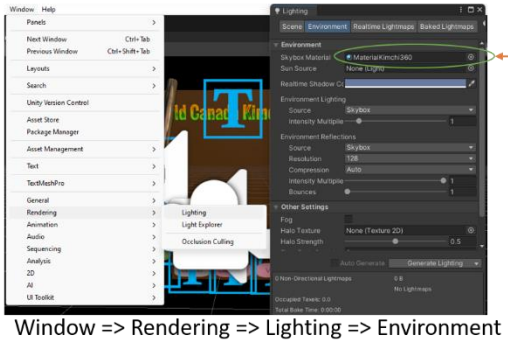
1.1 360 Degree Panoramic Video Play for Background

The first screen is as important as the first impression of a person. Since it is a promotional application, it needed to attract attention from the start. We needed a way to show the promotional video for the event that we made by making full use of the advantages of VR. While looking for a lively way to make full use of the 360-degree view, which is the biggest advantage of VR, I found out that a general video can be converted into a 360-degree view as the background and shown.

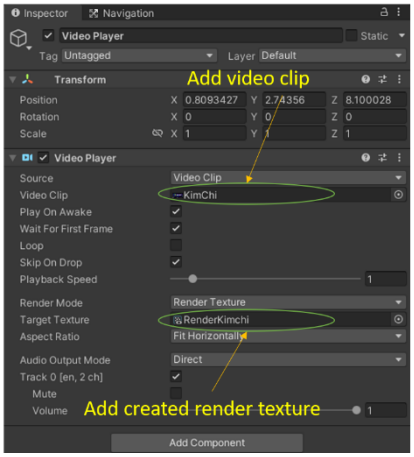
1) Video editing: Adobe Premier was used for making video clip for introducing. (*Assets/Audios/*Kimchi.mp4)

2) 360-degree video play – method 'b' was used.
   a. Using sphere: old method – mapping video clip onto inside the sphere surface. It requires several tricks.
   b. Using skybox: new method – easy




Window => Rendering => Lighting => Environment

Create new video player at hierarchy window



Comments:

(1) Higher resolution is required for better immersion.
(2) Any method to get more distance between user and videos?
(3) Better to add mosaic stadium footage instead of static image.

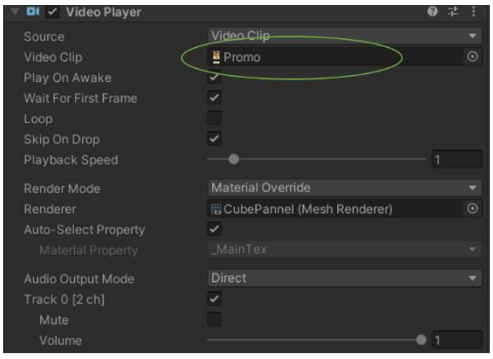
Screen monitoring

Figure 3 Panoramic video play

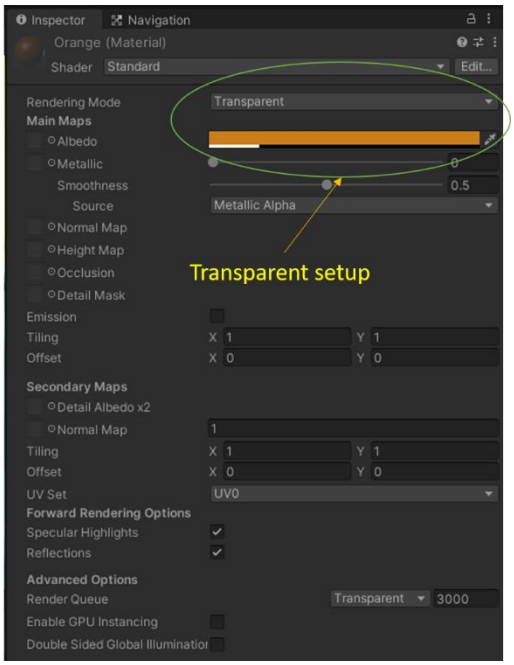
1.2 Main Screen: Half transparent screen video play

Modern marketing is all about splendid screen composition and constant tension. No matter how fresh the content, if it lasts more than 2 minutes, it is hard to avoid boredom. In order not to give a dull moment, a semi-transparent promotional video for the event is played in front of the camera. The sound of the music in the new video is not loud, so the sound was played together to heighten the tension.

1) Video editing: Adobe Premier was used for making video clip for introducing. (*Assets/Audios/*Promo.mp4)
2) Add 3D cube material as a screen, drag and drop footage, add material for color and shader.



Create 3D cube => Drag and drop video clip



Add material to 3D cube
for controlling color and shading

Comments:

(1) Appropriate screen size was found by trial and error.
(2) Appropriate transparent and color was chosen by trial and error.
(3) Muting the 2nd video would be better?

Screen monitoring

Figure 4 Half transparent screen

1.3 Scene Manager

1. Mini Game Menu

To enter the mini game from the main screen, 4 capsules were created vertically on the left side. The texture of each capsule is overwritten with an image, and the player can enter the game by hitting it with a stick attached to the left hand. The left stick is tagged with the name 'stick'. Since the player can only play the first mini game yet, the scene conversion script has been added only to the top capsule.

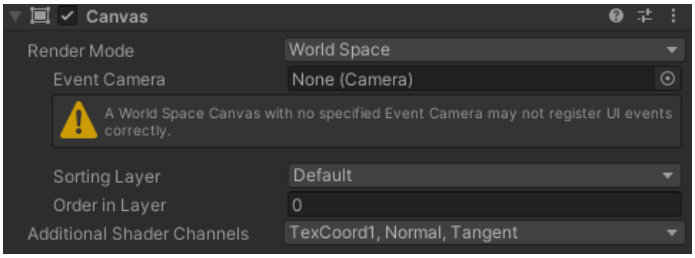
1) Add 3D capsules for selecting mini games.
2) Drag and drop images for texturing capsules. It creates materials automatically at ./materials folder. (*Assets/Pics/*chungha.jpg, tkd.jpg, hanbok.jpg, cabage.jpg)
3) Add canvas and text using TextMeshPro.
4) Add script to the first capsule for scene change. Targeting scene is Level1, and all scripts files are at *Assets/script* folder.

5) Codes for scene change

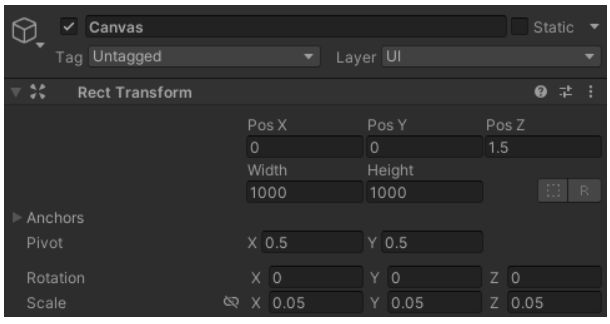
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class SceneCh : MonoBehaviour
{
    public string sceneName;

    private void OnCollisionEnter(Collision collision)
    {
        // If collision happens with 'stick', go to scene named 'sceneName'
        if (collision.gameObject.CompareTag("stick"))
        {
            SceneManager.LoadScene(sceneName);
        }
    }
}
```

Comments:

(1) The scene changes only when the capsule is hit by something tagged ‘stick’.
(2) Changing scene was not hard coded for flexibility.
(3) Add text is a bit complex. Render Mode of Canvas must be switched to ‘World Space’. Otherwise its location is hard to track.



(4) Even after switching world space, the distance of text and object is different. I think it is because Canvas is scaled to 0.05. Finding correct location for text was not easy.



(5) It would be nice to be able to add an effect that makes the menu translucent while background video is on. Then, not only it allows to skip the music by selecting menu, but the panoramic video will not be covered by the menu. Thus, the immersion will be better.

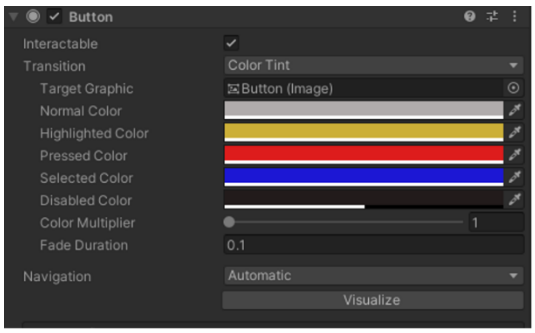
Screen monitoring

Figure 5 Mini game menu

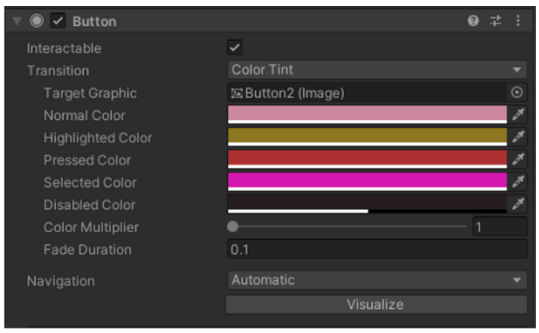
2. *Exhibition Hall Menu*

The button menu for entering the singer's exhibition hall is arranged horizontally at the bottom of the screen. The player can see the light beam from the right controller. If the player shoot the desired button with the trigger button, the player can enter the exhibition hall of the selected singer. When the button shot, the phrase 'Clicked' comes out for checking which button was selected. This feature was implemented by adding scripts to both the right controller and button menus. In the beta version, only 6band and Viviz exhibition halls can be visited.

1) Add 5 buttons. The colors of the buttons for male singers and female singers are different. The default colors for male and female signer are grey and pink, respectively. If the button is hot by the ray, grey changes to blue and pink changes to hot pink. And add text to canvas



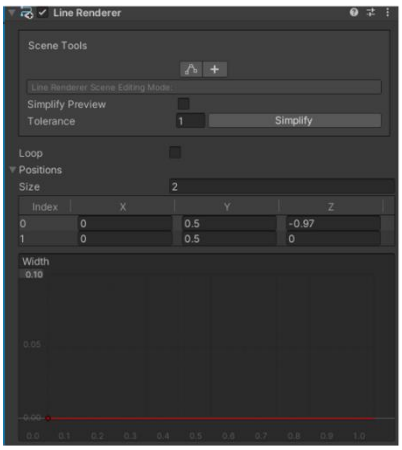
Button colors for male singer



Button colors for female singer

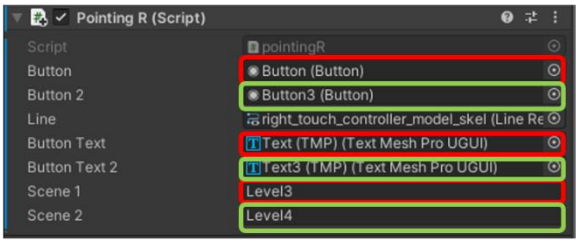
2) Add right_touch_controller_model_skel to RightControllerAnchor. Suggested model is included Oculus VR asset bundle.

3) Add line render and script to right_touch_controller_model_skel(Assets/script/pointing.cs). If ray hits button tagged ‘button’, it runs functions for 6band. If ray hits button tagged ‘button2’, it runs functions for Viviz. Since there are only two exhibition halls for 6band and Viviz, if the player shoots a male singer, the scene will move to 6band exhibition hall(Level3), and if the player shoots a female singer, it is set to enter the Viviz exhibition hall(Level4).



Determine ray width

for 6band
for Viviz

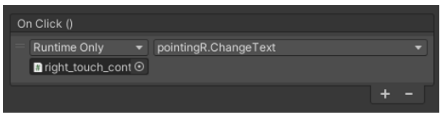


Button => Change Text(TMP) to ‘Clicked’ => Scene ‘Level3’

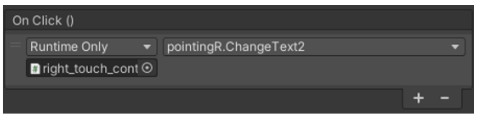
Button3 => Change Text3(TMP) to ‘Clicked’ => Scene ‘Level4’

Script was attached at **Appendix A**

4) Add functions to buttons. If buttons are clicked, run Text or Text2 functions. They are identical. They simply change texts to ‘Clicked’. Add right hand controller at On Click ( ) by triggering text switches.



Run Text if 6band was shot



Run Text2 if Viviz was shot

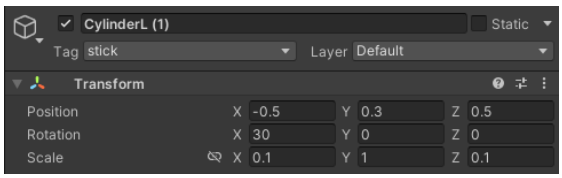
Screen monitoring

Figure 6 Exhibition hall menu

3. *Extra: Left Hand Stick and Dynamic Text*

Mini-games can be selected by hitting them with the left stick. Since there is a distance to the menu, the stick was moved in the z direction in advance, and the length of the stick was extended. The image texture was also applied to the stick by dragging and dropping the image. Additionally, when the menu is pressed, a sound effect is played. Also, the script for dynamic text was added on the top of the screen. Since the static text is obscured by the video around it, a dynamic element is added to the text as well.

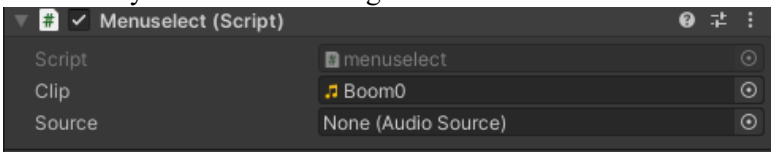
1) Add 3D cylinder to left hand. Size and offset values are adjusted by trial and error, so the player can hit the mini-game menu easily.



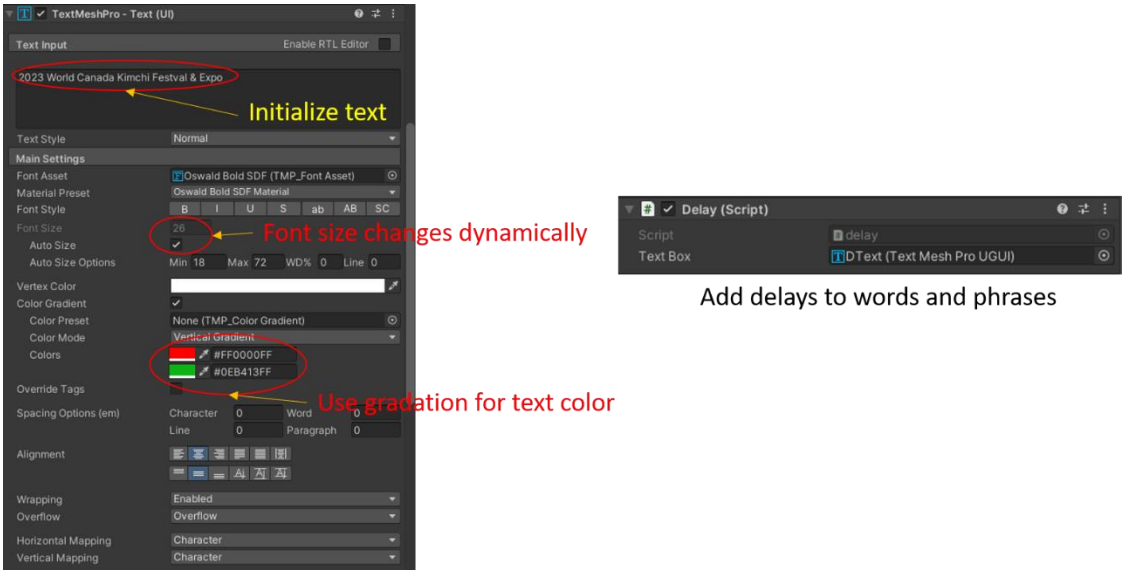
2) Drag and drop an image to the cylinder.
(Assets/Pics/Jisoo2.jpg)
3) Add script to cylinder for playing sound effect when mini-game menu is hit. The first capsule for the mini-game was tagged as ‘menu’ for enabling the script.
(Assets/script/menuselect.cs)

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

public class menuselect : MonoBehaviour
{
    public AudioClip clip;
    public AudioSource source;
    // Start is called before the first frame update
    void Start()
    {
        source = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {

    }
    void OnCollisionEnter(Collision collision)
    {
        // If the obj. tagged menu is hit, play 'clip'.
        if (collision.gameObject.CompareTag("menu"))
        {
            source.PlayOneShot(clip);
        }
    }
}
```

4) Add sound effect, Boom0 [6](Assets/Audios/Boom0.mp3) to the script. It allows to play Boom0 when the cylinder hit the mini-game menu.



5) Add a script for dynamic text on the top of the screen. It is a script to express the text dynamically at the top of the screen in time for the end of the video. The script was attached in Appendix B. (Assets/script/delay.cs)



Comments:

(1) The effect of text flying like in PowerPoint was suitable, but it was unable to be implemented due to technical limitations.

(2) The shape of the cylinder needs to be changed to be more aesthetically pleasing.

Screen monitoring

Figure 7 Dynamic text

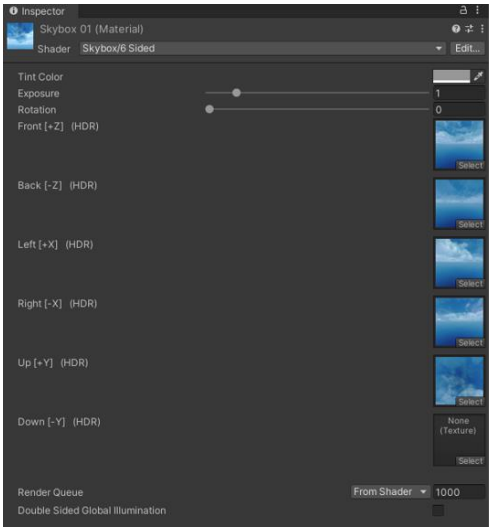
2. Mini-Game: K-pop Guardian

2.1 Background: Skybox, Terrain, Objects

The background of the game was set as a clear sky. Curvy terrain was added to avoid monotony as fans flocked to the stadium, and objects were added to deter various fans from approaching.

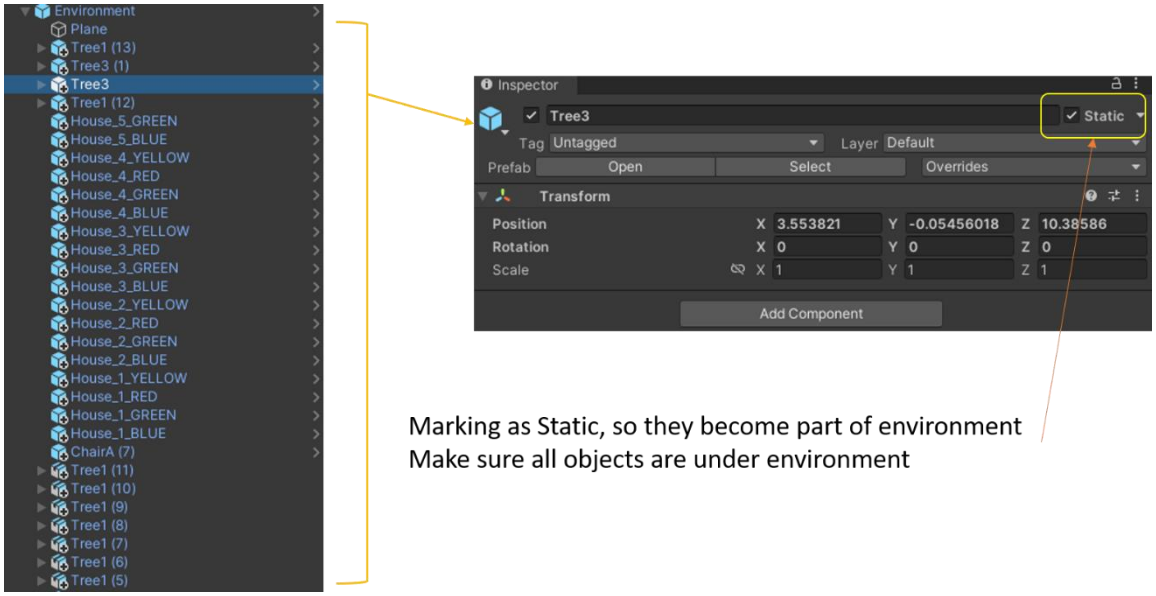
1. Skybox Creation, Terrain objects for environment

1) Drag and drop skybox material to the background for clear blue sky with downloaded assets. (Assets/Pics/Skybox1.mat)



6 side images for skybox

2) Add downloaded terrain to the scene by drag and drop. (Assets/Environment/environment.fbx)
3) Add downloaded objects to the terrain by drag and drop. (Assets/Environment/etc)
4) Create empty material for environment and add all objects and terrain under environment.



Marking as Static, so they become part of environment
Make sure all objects are under environment

2. *Other Objects*

1) Add large 3D cube behind camera and drag and drop stadium image. (Assets/Pics/mosaic-stadium.jpg)
2) Add another cube in front of stadium and add texture image to create a gate. (Assets/Pics/Portal.jpg)
3) Add 3D object indicating the target, so that the enemies can march toward the target. (Assets/Environment/etc/Models/GuardTower_3_lvl_snow_pal.fbx)

Comments:

(1) At first, high-definition wide terrain was used, but as an error occurred during the build process, the cause of the error could not be found.
(2) As it was rebuilt with a smaller terrain, several features were not properly updated yet.
(3) The surroundings need to be decorated well to look like a real stadium park.
(4) It would be nice if the stadium could be replaced with a more circular model.

Screen monitoring

Figure 8 Background environment

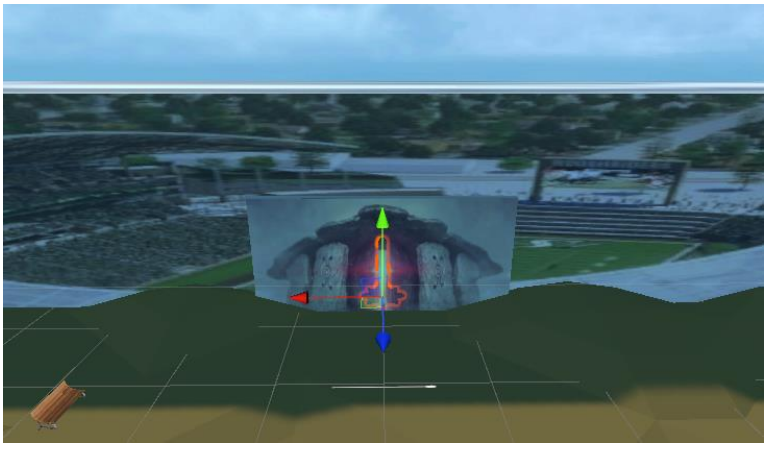


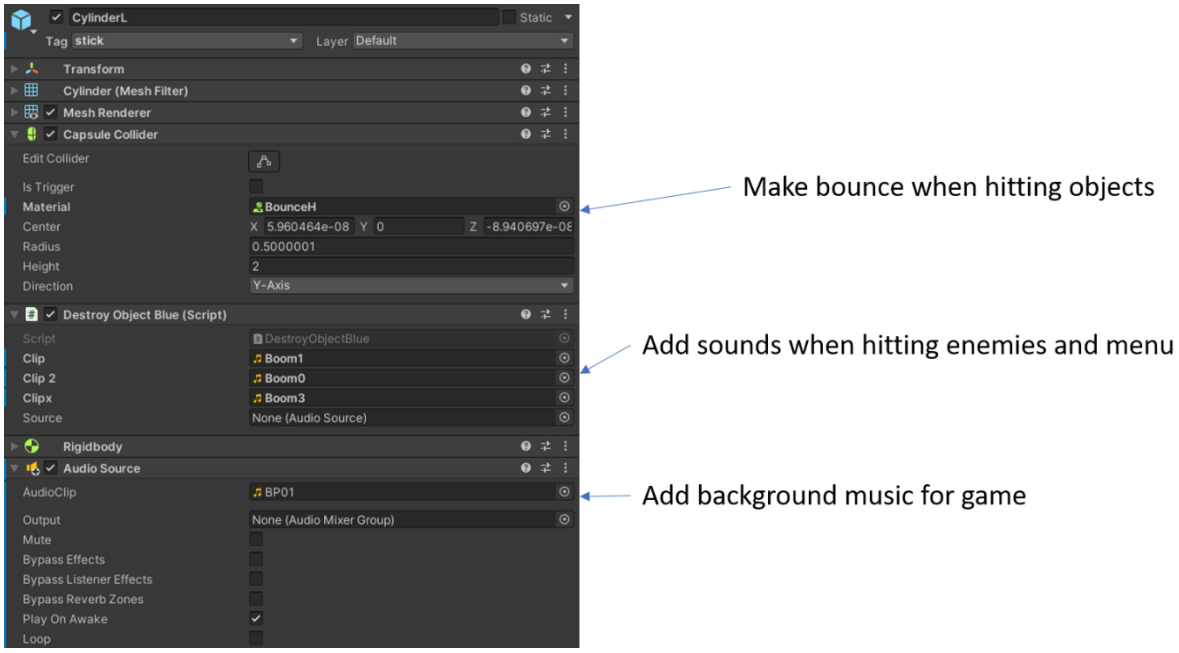
Figure 9 Stadium, gate, and tower

2.2 Effects: Crosshair, Bullet, Stick

Add effect for gun such as crosshair and bullet reflections. Attach the gun to the right hand.

1) Attach the gun to the right hand. The model, squirtgun_1 is included in Oculus package. Add gun script for 3 effects. Script was attached in Appendix C. Add three effects using gun at the hierarchy windows so it can be displayed during game.
(Assets/script/Gun.cs)
(Assets/Workshop/Effects/Explosion.prefab, MissBullet.prefab)
(Assets/Workshop/Crosshair_15.prefab)

2) Add 3D cylinder to the left hand and drag and drop the image. Add rigidbody for hitting enemies. Add sounds effects for hitting and add background music which is proper for shooting game.
(Assets/Pics/ TaeGeuk.jpg)



Comments:

(1) If aiming is on the air, crosshair is stuck with object.
(2) Find or make better effect files.

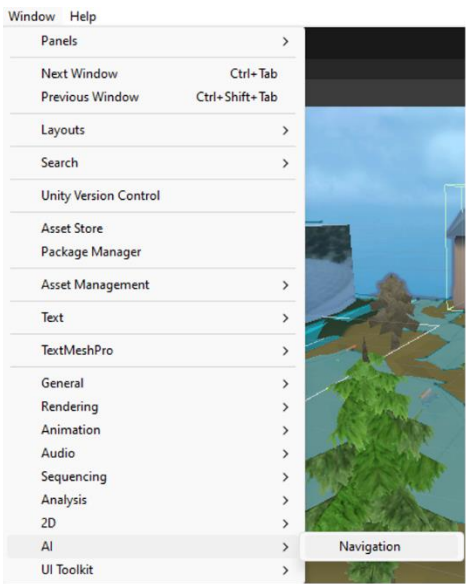
Screen monitoring

Figure 10 Add weapons

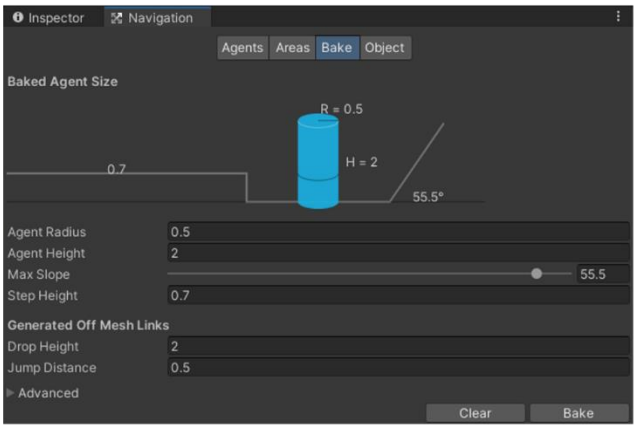
2.3 Add enemies

First of all, The navigating route was built using AI function in Unity. There are three different kinds of enemies. Male fan, Female fan, and Boss. They are all downloaded characters from Assets store. Add enemy spawning points for three characters.

1) Create navi mesh for finding route

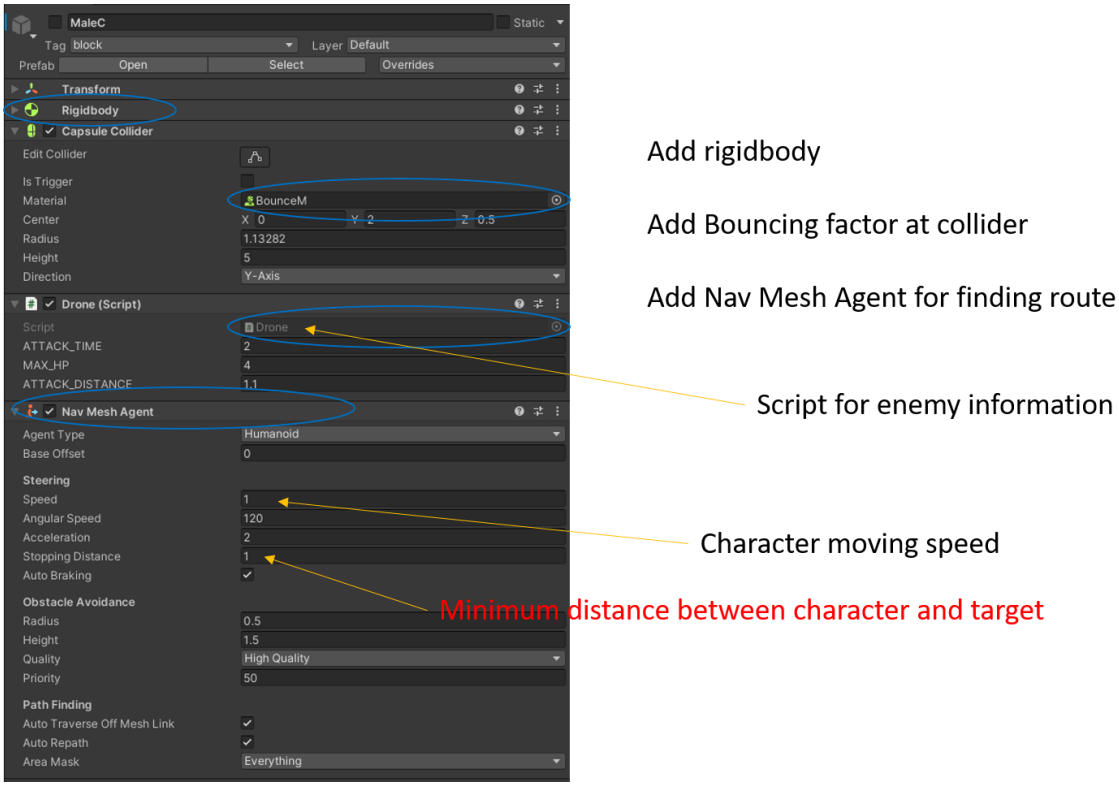


Click Window tap => AI => Navigation



Click Bake for creating route for movable area
It creates Navi mesh.
If Navi mesh is added to character, it gets route information

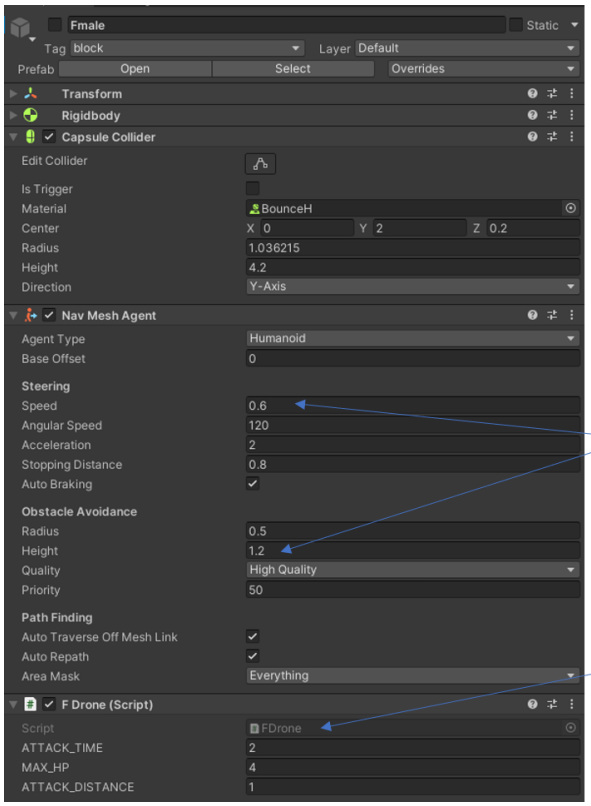
2) Add male fan character and add face image. Rigidbody, Collider, Nav Mesh Agent, and script was added. Script for male enemy was attached in Appendix D. It was edited from downloaded Drone.cs [7].
(Assets/Char/MaleC.prefab)
(Assets/Pics/boyface.jpg)
(Assets/script/Drone.cs)



3) Add script for spawning male fans.

```
using UnityEngine;
using System.Collections;
public class MSpawn : MonoBehaviour
{
	// object named drone, MIN_TIME, MAX_TIME
	public GameObject drone;
	public float MIN_TIME = 1;
	public float MAX_TIME = 5;
	// Use this for initialization
	void Start()
	{	// For giving delay
		StartCoroutine("CreateDrone");
	}
	// To use Coroutine
	IEnumerator CreateDrone()
	{
		while (Application.isPlaying)
		{	// Create Random time between MIN_TIM and MAX_TIME
			float createTime = Random.Range(MIN_TIME, MAX_TIME);
			// Delay for created random time
			yield return new WaitForSeconds(createTime);
			// Drone find position
			Instantiate(drone, transform.position, Quaternion.identity);
			// After creation, it drops tower's HP, so 1 hp for tower is recovered.
			Tower.Instance.Plus();
		}
	}
}
```

4) Add Female fan character and add face image. Rigidbody, Collider, Nav Mesh Agent, and script was added. Script for female enemy was similar with Appendix D. Only some constant values are different. Find details at the GitHub repository.
(Assets/Char/FmaleC.prefab)
(Assets/Pics/Face2.jpg)
(Assets/script/fDrone.cs)



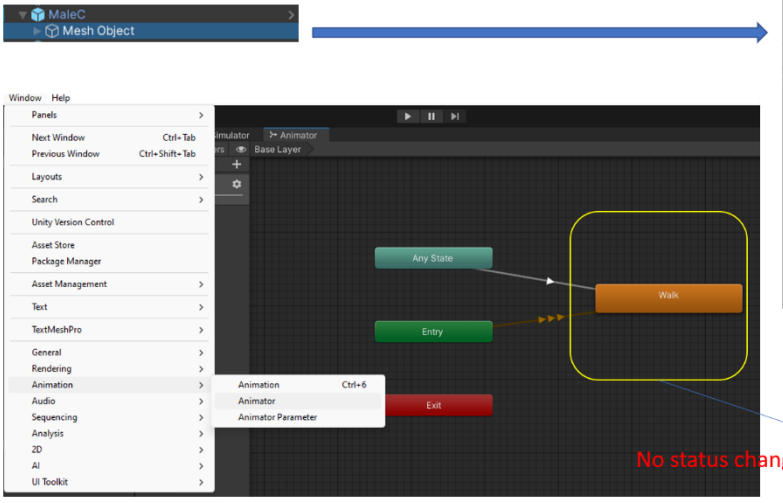
Similar setup with male fan

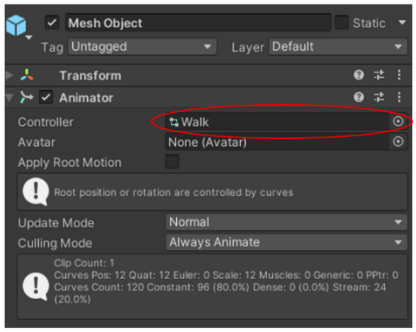
Changed variables
Speed
Height

Similar script with male fan

5) Add walking animation to characters

Male fan character





Walk is default animation

No status change. Character always walk

Window => Animation => Animator

6) Add downloaded boss character from Assets store. Edit animator only for ‘run’ at this time. Boss character runs. Add script for boss character. It is also similar with other enemies’ script.
(Assets/Char/AssGirlS/Characters/Sporty Girl/SportyGirl.prefab)
(Assets/script/Sboss.cs)

Comments:

(1) Only the state of the character's movement is used, but it needs to be changed.
(2) Whenever the enemy was spawn, it damages to the player’s HP. It was impossible to find the reason. Thus, 1 hp was added whenever the enemy was created.

Screen monitoring

Figure 11 Enemy characters

//////////////// From this point, I am still working on it since my previous version failed to build and crashed.

2.4 Add additional characters

2.5 UI

2.6 Attack and Game Over

3. Exhibition Halls: 6band & ViViZ – Scene Level3 & Level4

3.1. 6band: Stress resolving performance

Create a skybox background with the image of the venue to make it feel as if the player has come to see a real performance. A screen is installed in the concert hall and plays the singer's most famous hit song, 'I want to eat a banana'. The screen color is set to yellow to give a feel related to the song. Listening to music alone can get boring, so create objects you can hit so that they move with the music. The banana was created majorly. Carrots and tomatoes were created also to avoid monotony. However, make the bananas bigger so that they stand out.

1. Skybox Background

To make it look like a real concert hall, obtain an image of the concert hall from the assets store and implement it as a skybox.

1) Obtain dancing hall image from Assets store.
   (Assets/Pics/dancing_hall.jpg)
2) Create Skybox material and drag and drop the created material to the scene background.

Comments:

(1) Placing a few 3D objects for dancing hall will make it more realistic.
(2) It would be nice to place a 3D object in the shape of a person swaying to the music in the back.

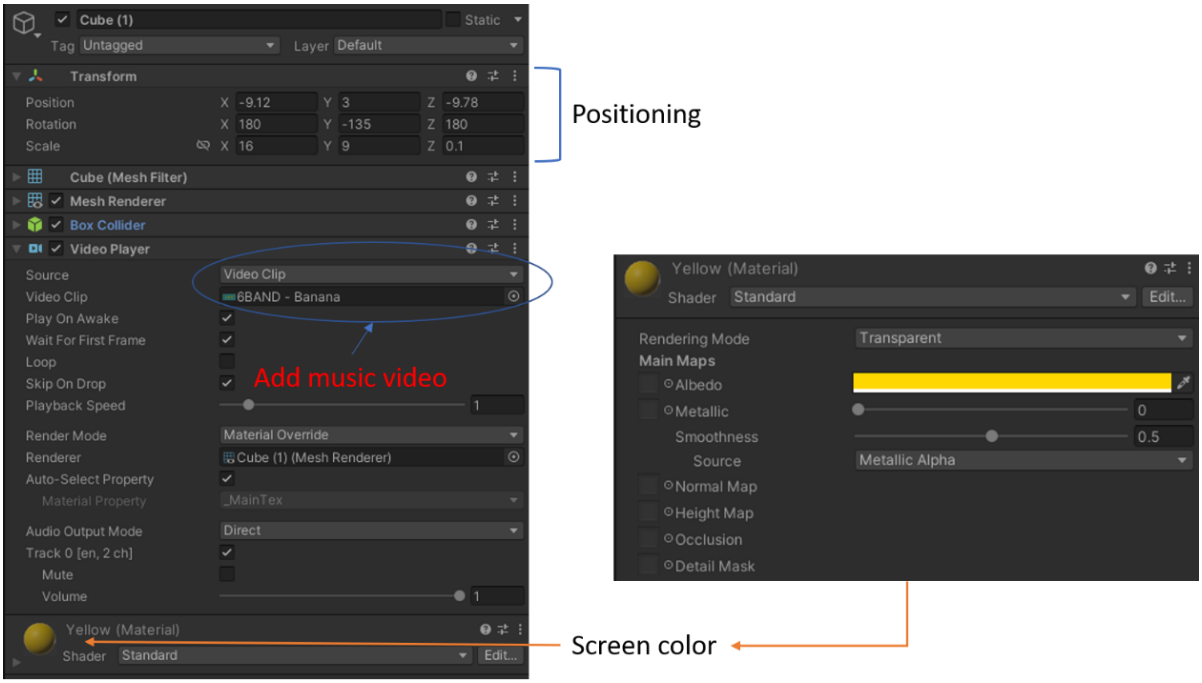
Screen monitoring

Figure 12 Skybox dancing hall background

2. *Flat Screen to play music video*

Make a large 3D cube to be used as a screen in front and play the 'I want to eat a banana' music video.

1) Create thin 3D cube.
2) Create material for yellow color and add it to cube
3) Drag and drop the video clip to the cube.
   (Assets/Audios/ 6BAND - Banana.mp4)



Comments:

(1) If the screen can flash to the beat…
(2) If only the screen could wriggle when the high notes came out.

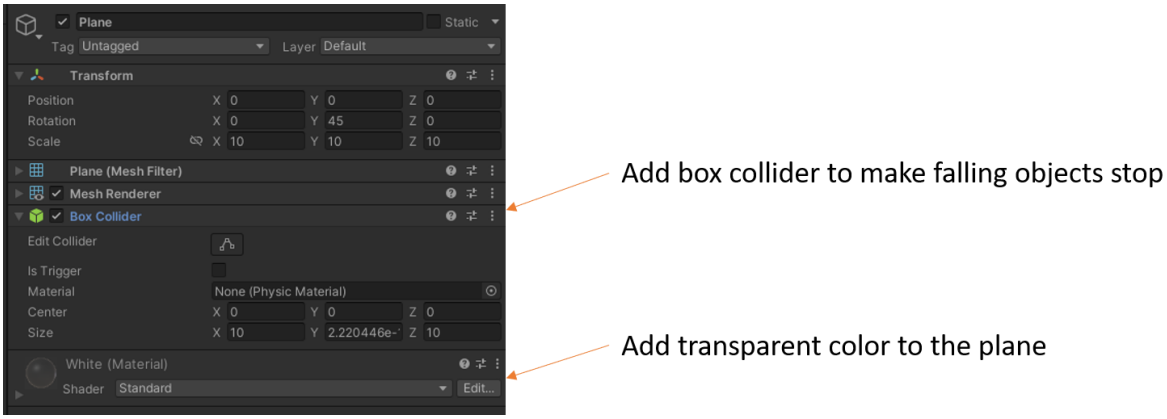
Screen monitoring

Figure 13 Yellow screen for 6band

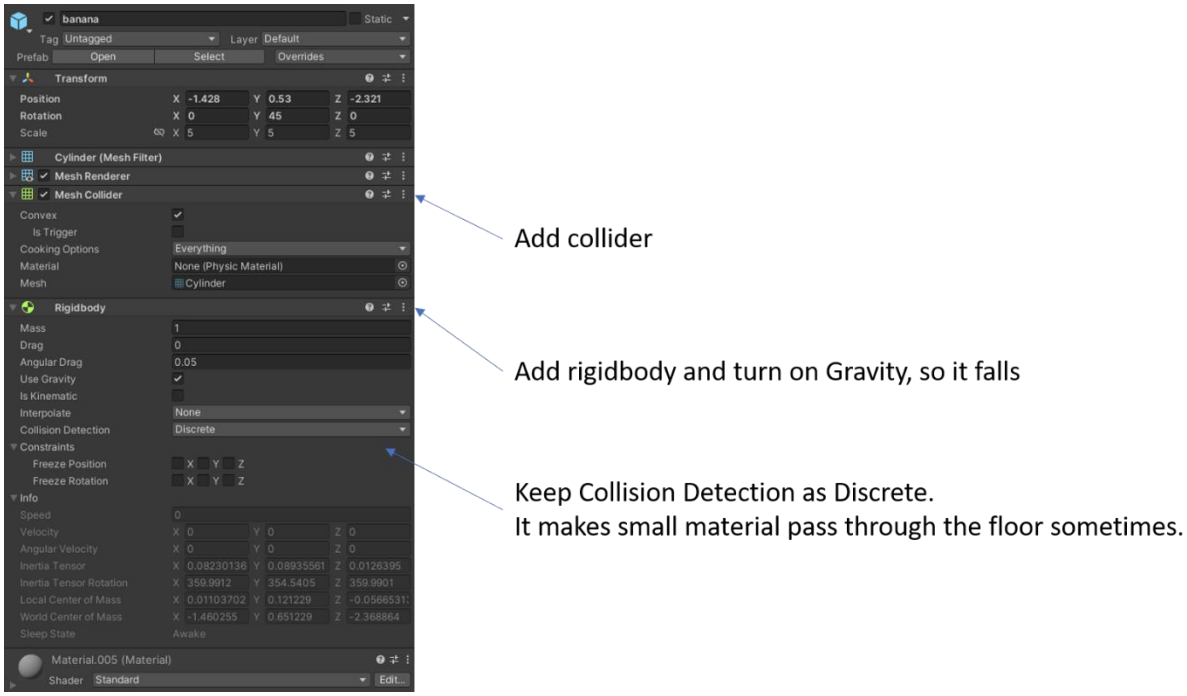
3. Add Fruits Materials

As the scene begins, set a large banana to fall in front of the player so the player doesn't get bored. The fruit continues to fall from the sky, continuing the dynamic feeling. A transparent bottom is installed on the floor so that the fruit stops at a suitable height for hitting with sticks later

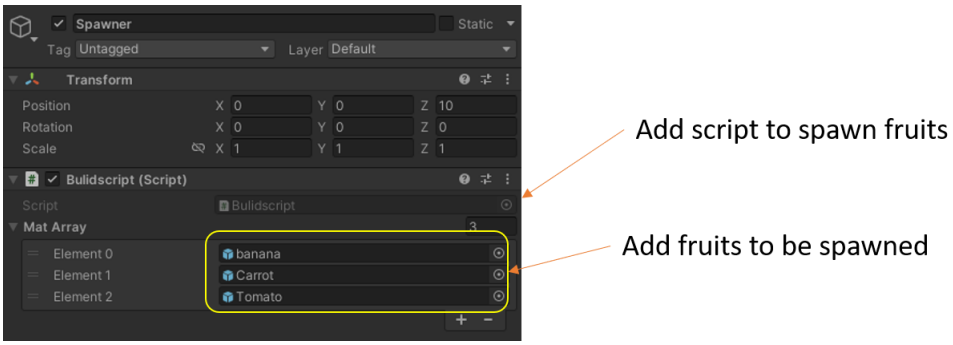
1) Create transparent floor



2) Download fruits objects from Asset store, and add one tomato, carrot, and banana. Make banana falls from the beginning to make it dynamic.
(Assets/Fruits)



3) Create empty material and add script (Assets/script/Buildscript.cs) for spawning fruits.




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

public class Bulidscript : MonoBehaviour
{
    // Save 2 obj. red and blue in array
    public GameObject[] matArray = new GameObject[3];

    // Start is called before the first frame update
    void Start()
    {
        // call Puff every 2 seconds
        InvokeRepeating("Puff", 1.0f, 0.5f);
    }

    // Update is called once per frame
    void Update()
    {

    }
    // Pop random blocks in random location and make them toward to player
    void Puff()
    {
        // Random # generator 0 or 2
        int newC = (Random.Range(0, 100) % 3);
        // Boundary for obj. spawning.
        float newx = Random.Range(-2.0f, 2.0f);
        float newy = Random.Range(10f, 20f);
        float newz = Random.Range(-2f, 2f);
        // Create one of 3 random objects
        GameObject CubeM = Instantiate(matArray[newC]);
        // Positioning created object
        CubeM.transform.position = new Vector3(newx, newy, newz);
        // Add rigidbody to object
        Rigidbody rb = CubeM.GetComponent<Rigidbody>();
        // Add velocity to downward so it can drop
        rb.velocity = new Vector3(0, -2f, 0);
    }
}
```

Screen monitoring

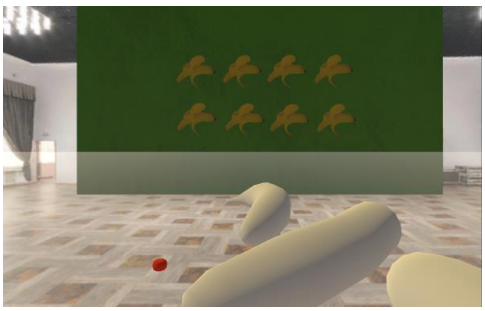
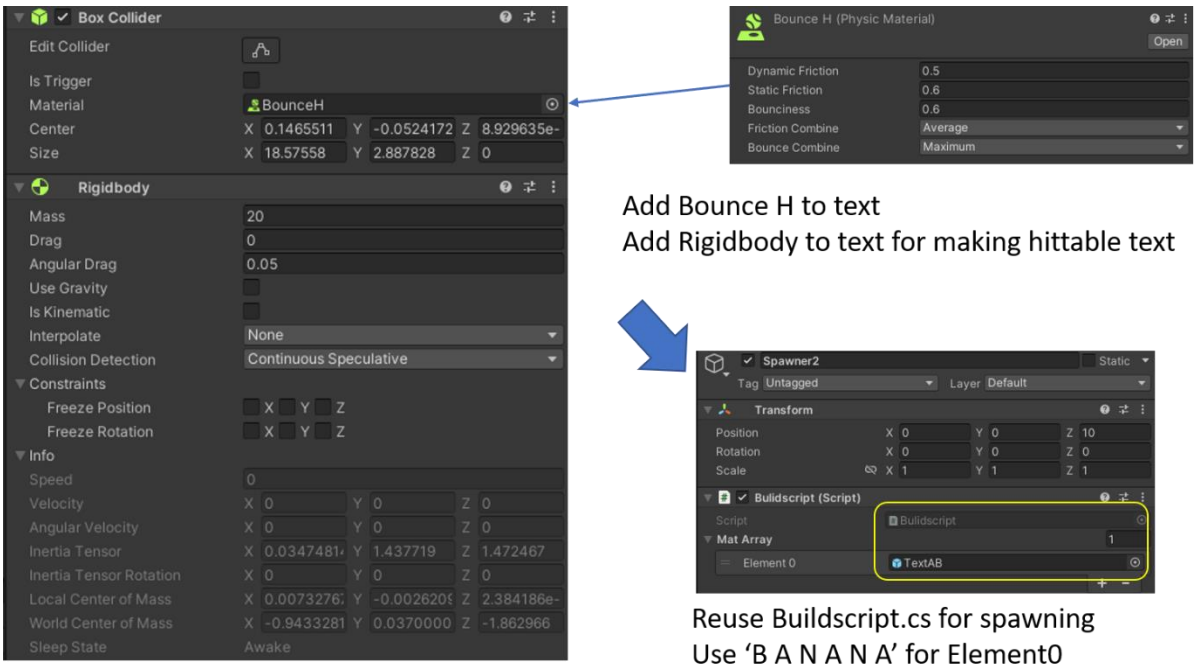


Figure 14 Spawn fruits

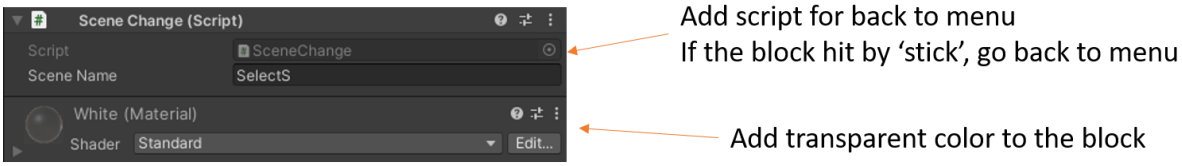
4. Decorate Sticks and Spawn Letters 'BANANA' to hit

Attach sticks to each hand so that they can hit objects to the music. Add some letters 'BANANA' and make the letters 'BANANA' fall gradually from the sky. Letters can also be hit with sticks. It enhances fun and resolves stress.

1) Add cylinder both hands and drag and drop singers' images to each cylinder. (Assets/Pics/6band.jpg, 6band2.jpg)
2) Add Canvas and change render mode to world space. Add button and texts, 'BANANA'. Add Rigidbody and bouncing component to text. And reuse Buildscript.cs for spawning 'B A N A B A' from the sky.



3) Add transparent 3D cube for 'back to menu' and add text on it.




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

public class SceneChange : MonoBehaviour
{
    public string sceneName;

    private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("stick"))
        {
            SceneManager.LoadScene(sceneName);
        }
    }
}
```

4) Add script to Canvas for emergency exit. If ‘x’ button is pressed, scene changes to main menu. (Assets/script/ScoreBoard.cs)

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

public class ScoreBoard : MonoBehaviour
{
    int count = 0;
    public TMP_Text textbox;
    public string scenemenu;
    // Start is called before the first frame update
    void Start()
    {
        textbox.text = "B A N A N A";
    }

    // Update is called once per frame
    void Update()
    {
        if (OVRInput.GetUp(OVRInput.RawButton.X))
         {
            SceneManager.LoadScene(scenemenu);
        }

    }

    void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Bblock"))
        {
            count--;
            textbox.text = count.ToString();
        }
        else if (collision.gameObject.CompareTag("block"))
        {
            count++;
            textbox.text = count.ToString();
        }
    }
}
```

Comments:

(1) Move menu box from front to back. When it was placed in front of the player, menu box was touched accidentally with the stick.

(2) Score board was added on ScoreBoard.cs so the score goes up whenever hit banana or letters. However, it didn’t work. It might have to be in Update( ) function.

Screen monitoring

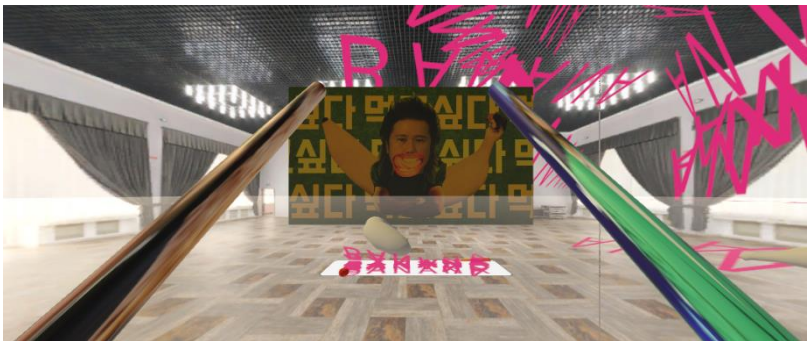


Figure 15 6band exhibition hall

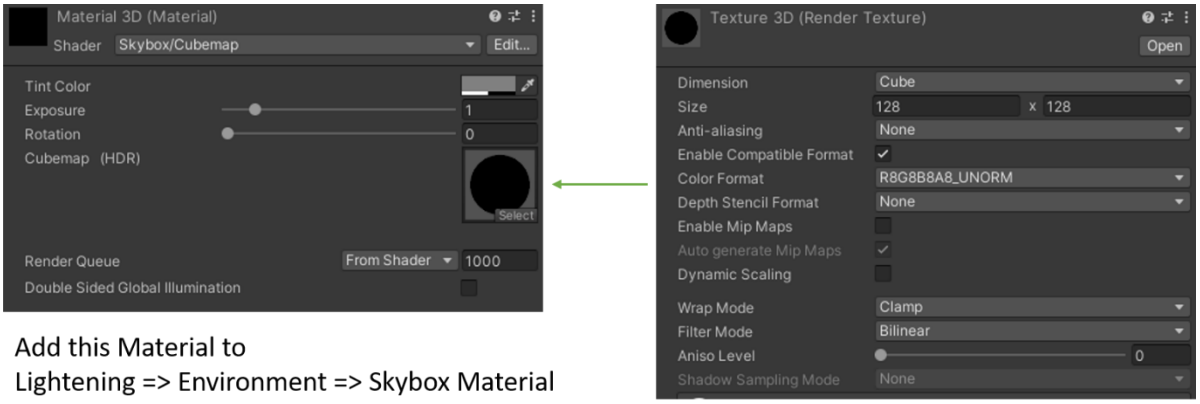
3.2. ViViZ: Exhibition Hall with Ecstasy

It is a setting where ViViZ is the only light in the universe, pitch-dark darkness. It feels like audience has moved to another dimension. The whole scene is surrounded by the black box. There is a structure confined by 20 spheres was formed in the middle of space. The light source comes out in 8 directions only from the inside of the structure. Inside, visitors can watch four different ViViZ music videos simultaneously in all directions. The goal is to create a sense of ecstasy.

1. Utter Black Skybox Background

The player should feel that ViViZ is the only one in the universe of utter darkness.

1) Add 3D black skybox for background.



Add this Material to
Lightening => Environment => Skybox Material

2) With 20 spheres, create a circular structure that is the same vertically, horizontally, vertically and horizontally
3) Add eight different color light sources are fired from inside the structure to 8 outward directions.

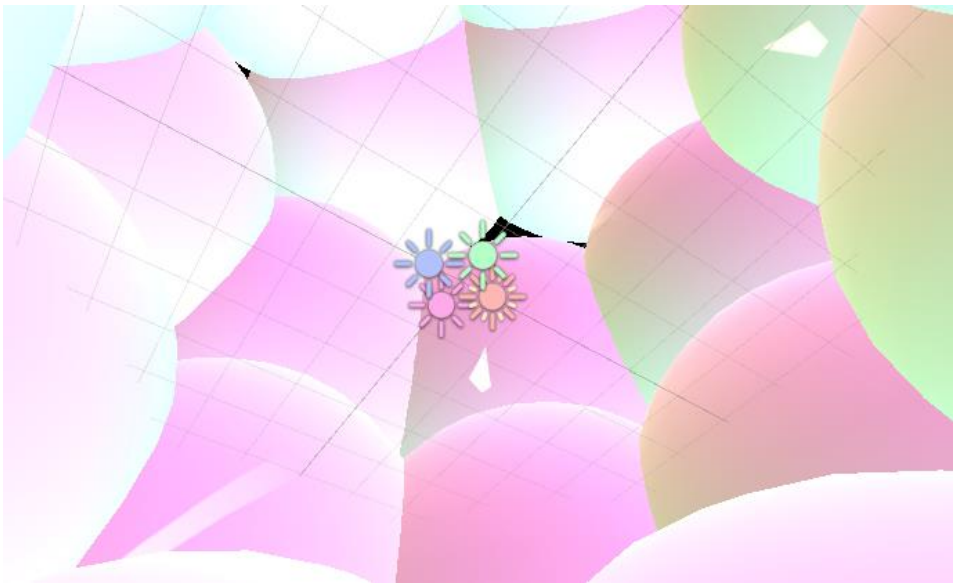


Figure 16 Scene capture with spheres and light sources

4) There are 4 stacks. Drag and drop different types of ViViZ music videos for each stack. Unmute only one music video.
5) Add Backto.cs script to any object. Backto.cs script allows to escape from this exhibition hall and go to main menu screen
(Assets/script/Backto.cs)

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

public class Backto : MonoBehaviour
{
    public string scenemenu;
    // Start is called before the first frame update
    void Start()
    {
    }

    // Update is called once per frame
    void Update()
    {
        if (OVRInput.GetUp(OVRInput.RawButton.X))
        {
            SceneManager.LoadScene(scenemenu);
        }
    }
}
```

Comments:

(1) How to play multiple music videos in succession. Too loud when all sounds are turned on at once.
(2) How to put a switch that turns music videos on and off.
(3) It would be more fun if we could let the player fly freely through space instead of moving on foot.

Screen monitoring

Figure 17 View from outside and inside structure

4. Functions: User Manual

1. How to start

The app starts from the menu screen. At any time, the player can select from the mini-game menu on the left or the exhibition hall menu below. The mini-game menu can be entered by hitting it with the left stick. You can enter the exhibition hall menu by shooting a ray pointed by the controller in your right hand with the trigger button. Currently, the first mini game and only the 6band and ViViZ exhibition halls are accessible.

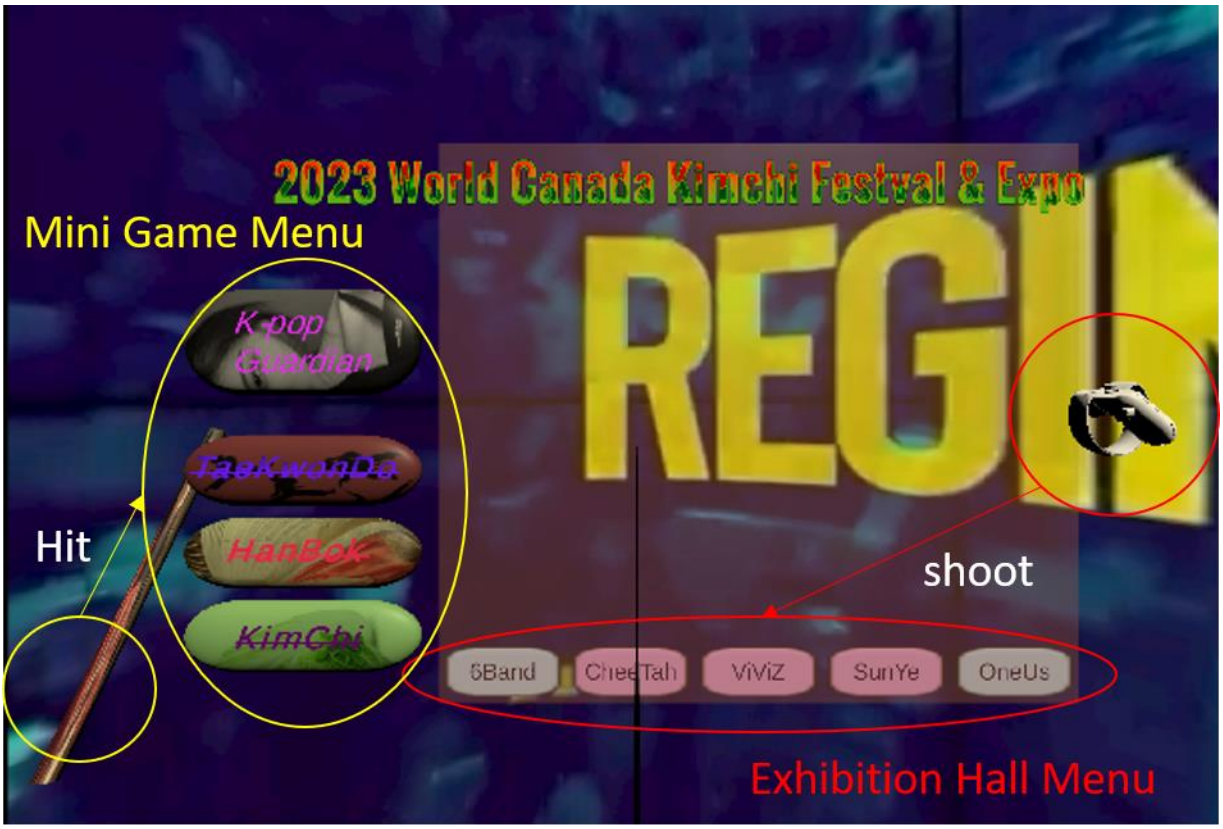


Figure 18 Instruction map for menu

2. Exhibition Hall

2.1. ViViZ Hall

Visitors can watch three-dimensional music video images while walking around the space at various angles. They feel pleasure only with your eyes, but there are no other interactive elements. Visitors can exit from this hall by pressing ‘x’ button at any time

Figure 19 ViViZ hall

2.2. 6band Hall

The sticks held in both hands can be wielded randomly to knock things around. Stress relief is the goal. The player can return to the main menu by hitting the letters that say menu on the back or pressing the 'x' button.

Figure 20 Front view and MENU block on the player's behind

3. Mini Game: K-pop Guardian

It is a game that protects the entrance of the stadium from rushing fans and uninvited celebrities until the singer enters. Enemies attack the player until the player dies. Genre is tower defense.

3.1. Characters

1. Enemies

There are 3 different enemies for this game. They are rushing to Stadium constantly. Each character has a different speed or ability to attack the player.

Figure 21 Enemies

2. NPC

They are standing at given position. They temporarily block the enemies but never move or attack others. This character is much larger than others. Sometimes it can be knocked down if really heavy object hits it.

Figure 22 NPC, K-pop star

3. K-pop Star

She shows up in the middle of game. She never dies. If the player touches her with right hand. The game will be over in success.

4. The Player

The player is invisible, but only HP is visible. HP slide bar exists on the near ground.

3.2. Weapons

They are two weapons. The player has a gun in his right hand and a stick in his left. When the player aims the gun, there is crosshair for the gun. The player can shoot by pressing the trigger button. The stick is used to push the characters.

Figure 23 Weapons and playing game image

5. Future Work

1. Short Term Work

1.1 K-pop Guardian

1) The map was so large that the process of rebuilding after the game crashed did not restore all of its previous functionality.
2) HP controls: enemy HP control – attack not working now, the player slide bar for HP
3) Bouncing control: cannot push back much with stick
4) Game over in happy/sad ending control

1.2 6band Exhibition Hall

1) Counter for counting the number of hits
2) More structures inside the hall
3) Adding animating dummy characters inside the hall

1.3 ViViZ hall

1) The player flies everywhere like swimming in the universe.
2) Music Video on/off control
3) Light color control

2. Long Term Work

1.1 More Mini Games

1) Taekwondo mini game
2) Hanbok fashion show
3) Kimchi making game

1.2 More Exhibition Halls for 3 other artists

1) Cheetah exhibition hall - Because she is a wrapper, the activity of finding and hitting words is being considered.
2) Sunye exhibition hall - TBD
3) Oneus exhibition hall - TBD

1.3 Exhibition Halls for June 25th

1) More ideas for 5 artists.
2) Beat drummer mini-game for Xdinary heroes since they are boy band.

Reference

[1] "McDonald's: Happy Goggles" McDonald, https://nordddb.com/case/mcdonalds-happy-goggles/.

[2] "Adidas Terrex: Delicatessen" Adidas, https://www.youtube.com/watch?v=FqH4E4tjxAk/.

[3] "World Kimchi Canada Festival & Expo." LIG Kimchi Ltd., www.lifeisgoodkimchifestival.net/.

[4] "Mosaic Stadium." Trip Advisor, www.tripadvisor.com/Attraction_Review-g155042-d15033971-Reviews-Mosaic_Stadium-Regina_Saskatchewan.html.

[5] "Tower Defender VR" Steam, https://store.steampowered.com/app/2054150/Tower_Defender_VR_Last_Adventure/.

[6] "Pixabay", https://pixabay.com/.

[7] https://www.gseek.kr/member/rl/studyRoom/studyRoomMain.do?courseSeq=2069&courseCsSeq=1&stuSeq=&subjSeq=1/.

Appendix A: Script for scene change with ray

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
using TMPro;
using UnityEngine.SceneManagement;

public class pointingR : MonoBehaviour
{
    public Button button;
    public Button button2;
    public LineRenderer Line;
    public TMP_Text buttonText;
    public TMP_Text buttonText2;
    public string scene1;
    public string scene2;
    // Start is called before the first frame update
    void Start()
    { }
    // Update is called once per frame
    void Update()
    {
        // generate a raycast
        RaycastHit hit;
        Ray ray = new Ray(transform.position, transform.forward);
        Physics.Raycast(ray, out hit);
        // Initialize ray position
        Line.SetPosition(0, transform.position);
        // shoot ray 20m forward
        Line.SetPosition(1, transform.position + (transform.forward * 20));
        // check if it hit anything
        if (hit.collider)
        {
            // If ray hits button
            if (hit.collider.CompareTag("button"))
            {
                // limit the ray position to hitting object.
                Line.SetPosition(1, hit.point);
                // run button
                button.Select();
                // Scene switch to scene1
                if (OVRInput.GetDown(OVRInput.Button.Any))
                {
                    // Allows switch text
                    ExecuteEvents.Execute(button.gameObject, new BaseEventData(EventSystem.current), ExecuteEvents.submitHandler);
                    SceneManager.LoadScene(scene1);
                }
            }
            // If ray hits button2
            else if (hit.collider.CompareTag("button2"))
            {
                // limit the ray position to hitting object.
                Line.SetPosition(1, hit.point);
                // run button2
                button2.Select();
                // Scene switch to scene2
                if (OVRInput.GetDown(OVRInput.Button.Any))
                {
                    // Allows switch Text
                    ExecuteEvents.Execute(button2.gameObject, new BaseEventData(EventSystem.current),ExecuteEvents.submitHandler);
                    SceneManager.LoadScene(scene2);
                }
            }
            // if ray is not shooting target button or button2,
            else
            {
                EventSystem.current.SetSelectedGameObject(null);
            }
        }
    }
    // After shot, text changes to 'Clicked'
    public void ChangeText()
    {
        buttonText.text = "Clicked";
    }
    public void ChangeText2()
    {
        buttonText2.text = "Clicked";
    }
}
```

Appendix B: Script for dynamic text

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
public class delay : MonoBehaviour
{
    public TMP_Text textBox;
    void Start()
    {   // call functions with delays
        textBox.text="";
        Invoke("DisplayText", 4);
        Invoke("DisplayText2", 10);
        Invoke("DisplayText3", 18);
        Invoke("DisplayText4", 20);
        Invoke("DisplayText5", 30);
        Invoke("DisplayText6", 32);
        Invoke("DisplayText7", 44);
        Invoke("DisplayText8", 46);
        Invoke("DisplayText9", 56);
        Invoke("DisplayText10", 57);
        Invoke("DisplayText11", 58);
        Invoke("DisplayText12", 59);
        Invoke("DisplayText13", 60);
        Invoke("DisplayText14", 62);
        Invoke("DisplayText15", 64);
        Invoke("DisplayText16", 67);
        Invoke("DisplayText17", 70); // End text motion with video ending
    }
    // Functions for texts to be displayed
    void DisplayText()
    {
        textBox.text="2023";}
    void DisplayText2()
    {
        textBox.text = "World";}
    void DisplayText3()
    {
        textBox.text = "2023 World";}
    void DisplayText4()
    {
        textBox.text = "Canada";}
    void DisplayText5()
    {
        textBox.text = "2023 World Canada";}
    void DisplayText6()
    {
        textBox.text = "Kimchi";}
    void DisplayText7()
    {
        textBox.text = "2023 World Canada Kimchi";}
    void DisplayText8()
    {
        textBox.text = "Festival & Expo";}
    void DisplayText9()
    {
        textBox.text = "2";}
    void DisplayText10()
    {
        textBox.text = "202";}
    void DisplayText11()
    {
        textBox.text = "2023 W";}
    void DisplayText12()
    {
        textBox.text = "2023 World";}
    void DisplayText13()
    {
        textBox.text = "2023 World Cana";}
    void DisplayText14()
    {
        textBox.text = "2023 World Canada Kim";}
    void DisplayText15()
    {
        textBox.text = "2023 World Canada Kimchi Fes";}
    void DisplayText16()
    {
        textBox.text = "2023 World Canada Kimchi Festival &";}
    void DisplayText17()
    {
        textBox.text = "2023 World Canada Kimchi Festival & Expo";}
}
```

Appendix C

```
using UnityEngine;
using System.Collections;
using UnityEngine.EventSystems;

public class Gun : MonoBehaviour {
        public Transform bulletImpact;
        public Transform explosion;
        ParticleSystem bulletps;
        ParticleSystem explosionPs;

    public Transform crossHair;

    Vector3 originSize;
        void Start()
        {
        originSize = crossHair.localScale * 3.2f;
        if (bulletImpact)
                        bulletps = bulletImpact.GetComponent<ParticleSystem>();
                if(explosion)
                        explosionPs = explosion.GetComponent<ParticleSystem>();
        }
        // Update is called once per frame
        void Update () {
                //if(Input.GetButtonDown("Fire1"))
                {
                        Ray ray = new Ray(transform.position, transform.forward);
                        RaycastHit hitInfo;

                        if(Physics.Raycast(ray, out hitInfo))
                        {
                crossHair.position = hitInfo.point;
                crossHair.forward = -1 * ray.direction;
                crossHair.localScale = originSize * hitInfo.distance;
                if (OVRInput.GetDown(OVRInput.Button.PrimaryIndexTrigger, OVRInput.Controller.RTouch))
                {
                    if (bulletImpact)
                    {
                        bulletImpact.up = hitInfo.normal;
                        bulletImpact.position = hitInfo.point + hitInfo.normal * 0.2f;
                        bulletps.Stop();
                        bulletps.Play();
                    }

                    if (hitInfo.transform.name.Contains("block") || hitInfo.transform.name.Contains("Bblock"))
                    {
                        if (explosion)
                        {
                            explosion.position = hitInfo.transform.position;
                            explosionPs.Stop();
                            explosionPs.Play();
                            explosion.GetComponent<AudioSource>().Stop();
                            explosion.GetComponent<AudioSource>().Play();

                        }
                        Destroy(hitInfo.transform.gameObject);
                    }
                    else
                    {
                        if (bulletImpact)
                        {
                            bulletImpact.GetComponent<AudioSource>().Stop();
                            bulletImpact.GetComponent<AudioSource>().Play();
                        }
                    }
                }
                        }
              }
        }
}
```

Appendix D

```
using UnityEngine;
using System.Collections;

// Needs AI Nav Mesh Agent
[RequireComponent(typeof(UnityEngine.AI.NavMeshAgent))]
public class Drone : MonoBehaviour {
	UnityEngine.AI.NavMeshAgent agent;
	// Tower location
	Transform tower;
	// Attack frequency
	public float ATTACK_TIME = 2;
	float attackTime = 0;
	// Maximum HP for the character
	public int MAX_HP = 3;
	[System.NonSerialized]
	// initialize variable hp
	public int hp = 0;

	// initialize attack distance to 1.
	public float ATTACK_DISTANCE = 1;
	// Use this for initialization
	void Start () {
		// find nav mesh agent component
		agent = GetComponent<UnityEngine.AI.NavMeshAgent>();
		// Set target to the object tagged 'Tower'
		tower = GameObject.Find("Tower").transform;
		// Go to the target
		agent.destination = tower.position;
		// Initial hp is Max
		hp = MAX_HP;
		// attacking freqency set as default
		attackTime = ATTACK_TIME;
	}

	void Update()
	{
		// If enemy is close to the target less than attack distance, start attack
		if(agent.remainingDistance <= ATTACK_DISTANCE)
		{
			attackTime += Time.deltaTime;
			if(attackTime > ATTACK_TIME)
			{
				attackTime = 0;
				// Add damage to Tower
				Tower.Instance.Damage();
			}
		}
	}
}
```