ICT SBA Report
PC Game with Kinect

Name : Lam Chun Ting Jeff

Class:6C

Class no: 13

Table of content

- Background information
- Details of the game
- Real implementation

Background information

Purpose and Aim

- designed for our school 50th anniversary celebration.
- Played in a booth

Therefore, need to

1. Have fun
2. Know more about our school history

- In order to meet the target of the celebration, I decide to let others know our school history through the game, not only having fun, which is an important element to a game. It will be placed in a game booth.

Basic info of game booth

- The group of people will come
  - Students
  - Teachers
  - Alumni
  - Primary school kids
  - Outsiders

  So the game needs to suit everyone in order to have fun and play

The length of the game booth

→2 days

Background of the game

- Winning in a competition using Kinect program , promote the technology of Kinect to others. (Somatosensory game)
- The game divide to two parts to meet the target
    1. Gaming Part (collecting balls)
    2. Challenges Part (answering a question)
- The idea of this game come from two games respectively
    1. Slenderman(get balls)
    2. TalesRunner(answer question)

More details can see P.15 for the details of game

Advantages of Somatosensory game

- Funny
- Player-friendly
- Realistic
- Cure disease -> healthy
- Various game can be made

→ Therefore it can meet the achieve the purpose mentioned.

How to make it(Game Engine)

- Decide to use Unity.
- A chart of game engine I have considered

Name of Engine	Support 3D or not	Quality of game can be made	Support Kinect or not	The level of difficultily
CryEngine	Yes	Highest	No (recent days)	Most Difficult
Unity	Yes	Acceptable	Yes	Difficult
Scratch	No	Lowest	No	Easy

How to make it(Game Engine)

- I decide to enhance the quality of the game. So 3D is needed for making a model of school. They can enjoy the view while collecting balls so Scratch is not considered.
- I decide to use Faast to connect with Kinect sensor to control the keyboard event. That's mean when I have a body movement, It just pressed a button of keyboard. So support of Kinect is not a major concern.
- I have compared both CryEngine and Unity, using CryEngine is much more demanding than Unity. As it is self-based assessment, I choose to use Unity to do similar things.

How to make it (Kinect Part)

- Since I use First Person Control(FPS) to play the game, keyboard event is enough for me to control the person movement. FAAST is a way to change body movement to keyboard event.
- There is a limitation of Kinect SDK. When I connect it directly with body movement, the FPS cannot be moved by unknown reasons. So Kinect SDK is not my expectation but FAAST does.
- FAAST is easy to use. Therefore I decide to use FAAST at the same time with Unity3D.

What elements of the game expected

- Visual space
- Barriers
- Colors
- Sounds
- Lighting
- Goals
- Mechanics
- Rules

From (Reference)

https://www.makegameswith.us/gamernews/298/5-basic-elements-of-game-design

The game's element I have

- Visual space --- The school model
- Barriers --- The time limit
- Colors --- The school model
- Sounds --- Clapping hands when ball get and the birthday songs when game end
- Goals --- Collect 10 balls and answer a question
- Rules --- Finish within the time limit

Others element should consider

- Short time – Prevent long waiting time
- Fun – a game should have
- Bring out messages within that gaming time – Knowing the history of school

The actual thing I have done

- Time limit (3 mins)
- Collecting balls with easy control

  → have a goal so it has fun
- The question ask at last

  →know the history of school

Therefore

Knowledge need from the lessons

→Programming skills(language and algorithm)

→Effective way to divide in Sub-problems

1) Game part (will mention in Details of game)

2) implementation part

2.1) Booth planning

2.2) Technical booth setting

Details of the game

Before start

- I follow the knowledge of writing program learnt from lesson

→ Dividing them into sub problems(easier to make the game)

1. Flow of the game (code)
    - →Getting balls
    - →Answering question

And at last linking them and debug

2. Control (code)
    - →Keyboard control
    - →Kinect control

Make the code step by step

Before start

3. Decoration (object)

→The settings of the virtual school (e.g. Area , rooms)

→Images of some board

4. GUI Interface (code & object)

→The instruction for the game

→Any other messages need to bring out or implement

5. Invisible object (object)

→ Detecting the movement of the FPS

The flow of the game (beginning)

- Tutorial and brief introduction

遊戲目標:(在3分鍾之內)

Game Target: (3 mins limited)

1.收集5個籃球與5個排球

1.Collect 5 basketball and 5 volleyball respectivily.

2.去儲物室回答一條問題

2.Go to the store room and answer a question.

3. 遊戲完結 Game Finished

The flow of the game(main)

- 1. Collect 10 balls --- 5 bask leyballs

- 2. Go to the store room to a ı

- 3. Game end

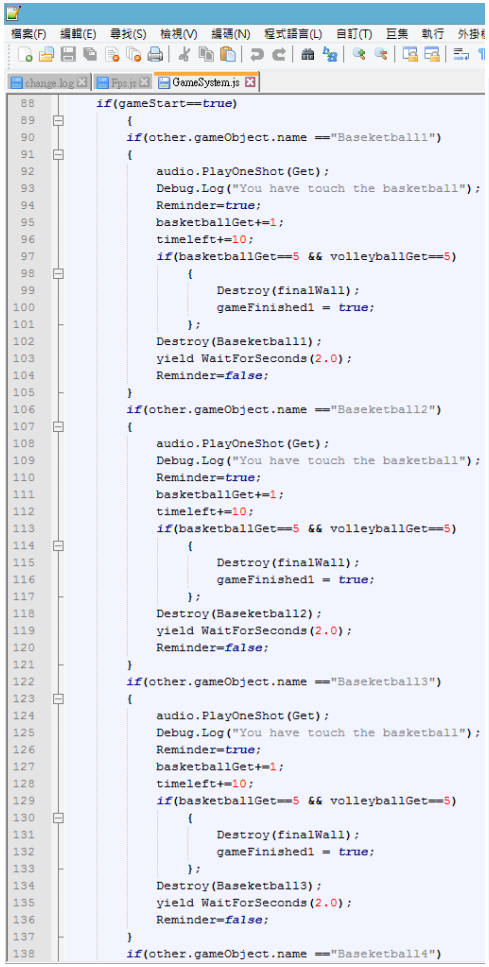
The reasons of this flow(beginning part)

- The tutorial is given before the game start with that 2 images.
  - As more accurate to know how to play other than only using words.
  - Let them know what's going on of how and what to play.

The reasons of this flow(main part)

- 1. Because it can help it achieve the aim and goal.
    a. Have fun
    b. Know more about our school history
- 2. Have fun through getting balls
    a. Can be suitable for both boys and girls, young and old(easy game)
    b. Can enjoy the view of the virtual school. (beautiful virtual space)
    c. Get back some memory if the alumni play it.
- 3. Knowing the school by answering question
    a. Through the question, we can know the answer of the question related to school.

Code of getting balls




```
}
if(other.gameObject.name =="Baseketball3")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the basketball");
    Reminder=true;
    basketballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==5)
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Baseketball3);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
if(other.gameObject.name =="Baseketball4")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the basketball");
    Reminder=true;
    basketballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==5)
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Baseketball4);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
if(other.gameObject.name =="Baseketball5")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the basketball");
    Reminder=true;
    basketballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==5)
```

```
}
if(other.gameObject.name =="Volleyball1")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the Volleyba
    Reminder=true;
    volleyballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Volleyball1);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
if(other.gameObject.name =="Volleyball2")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the Volleyba
    Reminder=true;
    volleyballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Volleyball2);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
if(other.gameObject.name =="Volleyball3")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the Volleyball");
    Reminder=true;
```

```
if(other.gameObject.name =="Volleyball4")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the Volleyball");
    Reminder=true;
    volleyballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==5)
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Volleyball4);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
if(other.gameObject.name =="Volleyball5")
{
    audio.PlayOneShot(Get);
    Debug.Log("You have touch the Volleyball");
    Reminder=true;
    volleyballGet+=1;
    timeleft+=10;
    if(basketballGet==5 && volleyballGet==5)
        {
            Destroy(finalWall);
            gameFinished1 = true;
        };
    Destroy(Volleyball5);
    yield WaitForSeconds(2.0);
    Reminder=false;
}
};
```

Flow of getting balls

The ball touched

↓

Appear the reminder of get(See GUI p.44)

↓

The total ball get +1

↓

10 second for bonus

↓

Determining all balls got

↓

Destroy the volleyball or basketball(Make it disappear)

↓

Hold for 2 seconds(For message only)

↓

Reminder of get disappeared

Explanation

- Mainly in OnTriggerEnter

→When collider of balls touched the player, then the incident of the flow appeared

→Messages need to write in OnGUI (see P.44) because in this function we cannot make the GUI appeared

Code of answering question

C:\Users\user\D

檔案(F) 編輯(E) 尋找(S) 檢視(V) 編碼(N) 程式語言(L) 自訂(T) 巨集 執行 外掛模組(P) 視窗(W) ?

change.log | Fps.js | GameSystem.js

```
var HB:AudioClip;
var GetImage:Texture;

function Start(){
    type= Random.Range(1,7);
    if (type > 3)
        place = false;


function OnTriggerEnter(other:Collider){
    if(gameStart1 == true)
    {
    if (other.gameObject.name == "Green")
    {
        if (place == false && gameFinished==false)
        {
            Debug.Log("Good");
            gameFinished = true;
            ans= true;
            audio.PlayOneShot(HB);
        }
        else
        {
            ans = false;
            timeleft = timeleft/2;
        };
    };
    if (other.gameObject.name == "Red")
    {
        if (place == true && gameFinished==false)
        {
            Debug.Log("Good");
            gameFinished = true;
            ans = true;
            audio.PlayOneShot(HB);
        }
        else
        {
            ans = false;
            timeleft = timeleft/2;
        };
    };
    }
    if(gameStart==false)
    {
        if(other.gameObject.name =="StartingWar")
        {
            Debug.Log("You have enter the wall");   //When you exit the wall and s
            gameStart= true;
        };
```

```
function OnGUI(){
    if (gameStart == false)
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10 , 512 , 256), "<size=20>The game will start after you leave the room</size>");
    if (gameFinished== false)
    {
        targetBasketball.text = "Basketball:"+basketballGet+"/5";
        targetVolleyball.text = "Volleyball:"+volleyballGet+"/5";
    };
    if (gameFinished1 == true&& gameStart1 == false)
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 1980 , 1072), "<color=yellow><size=30>Please go to the store room for next stage</size></color>");
    if (gameStart1 == true&& gameFinished == false&& ans ==true)
    {
        if(type==1) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2- Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1964  The green is:1946</size></color>");
        };
        if(type==2) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year New Block buil?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1975  The green is:1985</size></color>");
        };
        if(type==3) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1964  The green is:1946</size></color>");
        };
        if(type==4) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1949  The green is:1964</size></color>");
        };
        if(type==5) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2- Screen.height/10 , 512 , 256), "<color=black><size=30>In which year New Block buil?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1980 The green is:1975</size></color>");
        };
        if(type==6) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256), "<color=black><size=30>The red is:1949  The green is:1964</size></color>");
        };
```

```
if(ans == false&& gameFinished == false && gameOver == false)
{
    GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 , 512 , 256), "<color=Red><size=50>You answer wrong!</size></color>");
    GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2+50 , 512 , 256), "<color=Red><size=50>Please try again!</size></color>");
}
```

Flow of answering the question

Start(Red Part)

1. Random the type of question
2. By the type of question, know red or green is the right place to enter.

When enter to store room(Green Part)

1. Appear the type of question

Answer the question(Red Part)

1. Determining if it enter the right area(green or red)

Correct
Then game end

Wrong
Then answer again

(Blue Part) messages appear

Explanation(red part)

- Is determined if answering correct or not
- Written in the function OnTriggerEnter

→The red and green area have an invisible object(See P.52)

→When it enter to it, I can know if it is answered(enter to the collider)

→Variable "place" is determining the right answer is in green or red

- Variable "ans" is considered if it answer write or not

→The blue part will react to this variable and appear message

Explanation(Green part)

- The question asked(Just appear one of them)
- Written in the function OnGUI(a funtion show GUI)
- 3 different types , for each 2 different set of answer

Explanation(Blue part)(Details see P.49)

- A message of answering wrong

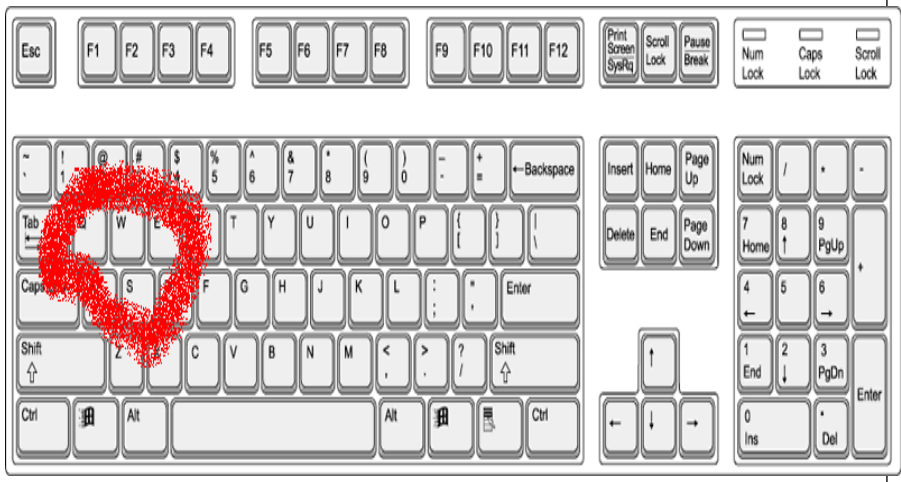
The whole flow (video explanation)

GameFlowEplanation.mp4

How to control

Using hands to control

Using keyboard of WSQE



The reasons of this control

- For the Kinect one, it is easily learnt by everyone.
  - 2 others method have been tested.
    - 1. Using left hand to control movement and using the shoulder to control the viewing angle
    - 2. Using right hand to control movement and left hand to control the view angle

	Method 1	Method 2
Learnt time	Long	Medium
Difficulty to control	Difficult(Too exhausted)	Acceptable
Bug exist	No	Yes(Duel to hands cross over)

The method using now is relatively better that short learnt time, Easy to control and not even a bug exist

The reasons of this control

- For the keyboard one
- Why not WASD?

  A and D have already been occupied by the Unity FPS controller so I cannot use it as changing view angle.

Details can see the FPS code of P.34

QE is just near by the WS so it can easily recognize. As a result, QE is used for changing the angle.

The reasons of 2 kinds of control

- Main aim is to use Kinect to play the game
- The reasons of using keyboard as well
  - If the Kinect is broken down accidently, keyboard can be a way to play game for the booth
  - It is easy for me to do testing since I need not to use lots of energy to do it.

FPS controller

- FPS stands for first person control.
- Reasons of using FPS rather than Third person control.
  - Third person control will affect the player enjoying the view of the virtual school.
  - Third person control is not easy for me to design one since it is used another technic called 3D mesh which is a kind of limitation.

FPS controller

- Unity 3d has provided a sample of FPS control code and I just used it.
- Reasons
  - It can save time and focus on building the school and game design.
  - The similar coding made I have tried cannot be as realistic as the one provided.

  However , some new code is added that written by me for moving the viewing angle.

The reasons of adding new code

- The original code is used mouse to control view angle rather than keyboard

→mouse is not required and only need a keyboard instead

- I now use QE for changing the view angle because the mouse cannot be controlled with FAAST by unknown reason. So it replace mouse by QE and it works.

code of FPS(Unity provided)[3 section]

When “horizontal button” clicked ⟶ Move along y axis

When “vertical button” clicked ⟶ Move along x axis

** Depends on left or right , top or left(Unity code)

The viewing angle is controlled by mouse intensively

→ When mouse moved , same movement

When “ space bar click” ⟶ Jump

 = used

 = not used

code of FPS(Added)

When “Q” clicked → Camera moved to right

When “E” clicked → Camera moved to left

→ To replace the part of Unity provided code

Source code of FAAST

When both hands are above the torso → "W" clicked (FPS moved forward)

When both hands are under the waist → "S" clicked (FPS moved backward)

When left hand move left to the left shoulder → "Q" clicked (Camera moved to left)

When right hand move right to the right shoulder → "E" clicked (Camera moved to right)

**** FAAST cannot control the movement of mouse in a software which should moved the mouse accordingly to the control by unknown reason, so QE is needed**

→ Therefore, the FPS can be controlled by both keyboard and Kinect sensor

Decoration

- First, the setting

Decoration

Reasons

→This setting is similar to our school.

→The way of the court and space set is similar.

Decoration

- Some images of the virtual school

Decoration

- The virtual school is based on this image
- Making it similar, so people(especially alumni) can get back the memory, can also know more about the school.

GUI interface

Purpose

→GUI is to show some messages to the player

→Written in the function OnGUI

→That can show display the words and graphic.

Overall usage

- There are 4 GUI graphics
- There are 7 GUI label in total
- There is 1 GUI button
- Also 1 GUI text(not written in OnGUI)

4 GUI graphic

- Tutorial(2)

→Show instruction to the player

遊戲目標:(在3分鍾之內)

Game Target: (3 mins limited)

1.收集5個籃球與5個排球

1.Collect 5 basketball and 5 volleyball respectivily.

2.去儲物室回答一條問題

2.Go to the store room and answer a question.

3. 遊戲完結Game Finished

- Map

→Show what exactly the virtual school is

- Ball got

→Tell they have get the ball

7 GUI label

C:\Users\振霆\Desktop\SBA\SBA\New Unity Project\Assets\ScriptofCm\GameSystem.js - Notepad++

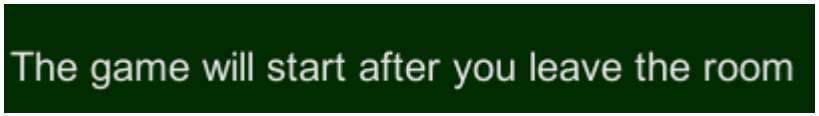
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

GameSystem.js

```
function OnGUI(){
    if (gameStart == false)
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10 , 512 , 256), "<size=20>The game will start after you leave the room</size>");
    if (gameFinished== false)
    {
        targetBasketball.text = "Basketball:"+basketballGet+"/5";
        targetVolleyball.text = "Volleyball:"+volleyballGet+"/5";
    };
    if (gameFinished1 == true&& gameStart1 == false)
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 1980 , 1072), "<color=yellow><size=30>Please go to the store room for next stage</size></color>");
    if (gameStart1 == true&& gameFinished == false&& ans ==true)
    {
        if(type==1) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2- Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1964  The green is:1946</size></color>");
        };
        if(type==2) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year New Block built?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1975  The green is:1985</size></color>");
        };
        if(type==3) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1964  The green is:1946</size></color>");
        };
        if(type==4) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1949  The green is:1964</size></color>");
        };
        if(type==5) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2- Screen.height/10, 512 , 256), "<color=black><size=30>In which year New Block built?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1980 The green is:1975</size></color>");
        };
        if(type==6) {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 - Screen.height/10, 512 , 256), "<color=black><size=30>In which year Carmel published?</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/10*9 , 512 , 256)  "<color=black><size=30>The red is:1949  The green is:1964</size></color>");
        };
    }
    if(gameFinished == true)
    {
        GUI.Label(Rect(Screen.width/2-Screen.width/6 , Screen.height/2-120 , 1920 , 1080), "<color=yellow><size=70>Happy Birthday to Carmel!</size></color>");
        if(GUI.Button (Rect(Screen.width/2-Screen.width/13 , Screen.height/2 , 300,40),"Restart Game"))
            Application.LoadLevel("Instruction");
    };
    if(gameOver == true)
    {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2-100 , 512 , 256), "<color=yellow><size=50>Game Over!</size></color>");
        if(GUI.Button (Rect(Screen.width/2-Screen.width/11 , Screen.height/2 , 300,40),"Restart Game"))
            Application.LoadLevel("Instruction");
    };
    if(ans == false&& gameFinished == false && gameOver == false)
    {
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2 , 512 , 256), "<color=Red><size=50>You answer wrong!</size></color>");
        GUI.Label(Rect(Screen.width/2-Screen.width/10 , Screen.height/2+50 , 512 , 256), "<color=Red><size=50>Please try again!</size></color>");
    }
    if (Reminder == true)
        GUI.Label(Rect(Screen.width/2-Screen.width/20 , Screen.height/2 - Screen.height/10 , 512 , 256), GetImage);
```

7GUI Label

Orange Part



→Tells how the game will start

Grey Part

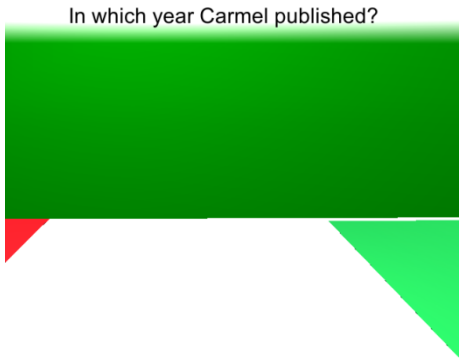
→Show how many balls have gotten and what the target

Yellow Part

→To show what the player should do after all the ball collected

→An information of next stage

7GUI Label



- Green Part

→The question and the answer of the last stage

→6 possibilities, that random when the game start

→ then it will display accordingly

- Blue Part

Happy Birthday to Carmel!

→ A message of winning the game(50th celebration)

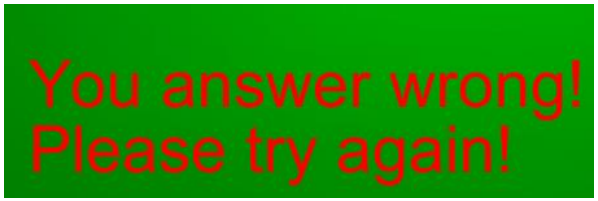
7GUI Label

Purple part

→When times-up, the game over will display

(show player is lose)

Pink part

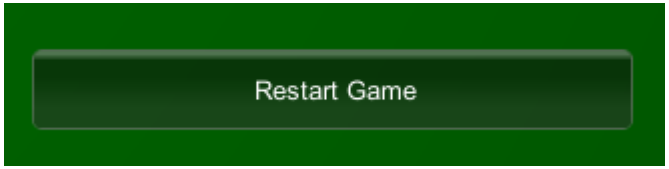


→ If the player answer wrong, this message will appear

GUI Button

- It is the only one button used in the game

→Used to restart game



1. When player win
2. When player lose

It will go back to the beginning of the game

GUI text

- The only used of this kind is in timer

→Written in update

→ Update continuously (1 second decrease 1)

Will add 10 seconds when ball get

→Limit the time for the player to play the game

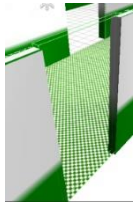
→Notice the time left

Object of the game(invisible)

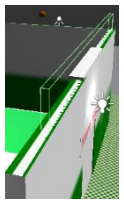
- They are used to be detecting the player

→Boolean variables will change because of it

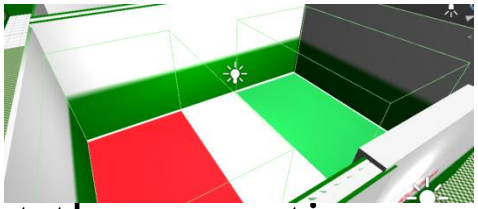
→The starting wall (Can get the ball after leaving the room)



→The second stage wall (Knowing that the player has entered the second stage)



→The answering rectangle (Know which answer have they chose)



As a result, different GUI will not appear at the same time because of the Boolean changed by invisible object.

Debug

- It can really meet my expectation after finishing it

→Only one bug existed in testing

→Store room can enter even not all balls are collected

→It fixed because of logic error (the comparison of ‘or’ and ‘and’)

→It works properly

→It will do testing and debug after any changes made for 2 times.

Testing

- The testing have made after some changes made for 2 times and 20 times after the game done

→20 times test is by playing the game by myself and also play and receive comment from the player(classmates) I invited before it can be played by everyone

- Some testing made and found out

→The random of question once get the same result three times(however, it still okay of the randomize)

→Some of the ceiling may have a large contrast, but in fact it does not really affect the game(Art problem)

Limitation of the game

- The balls cannot be randomly spawn

  a) Unity cannot spawn the balls in different places randomly which can only spawn in a fixed pattern

- The words color is little bit not sharp

  a)As the background (virtual school) have different color, the color of words cannot change with the detect of the color changes of background

  → Not easy to read some instruction of GUI words

- The GUI cannot suit in different monitor size(Will have displacement)

Further improvement

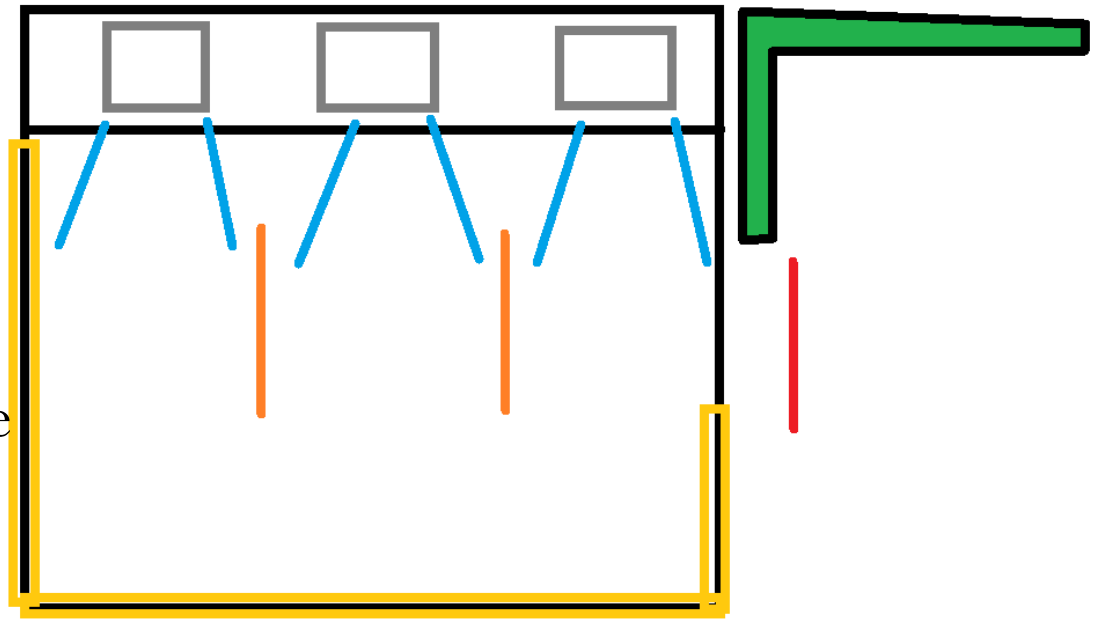
- Ball randomly spawn with 10 possibility
- The virtual school can be designed really likes the actual school even the furniture placement of the playground and classroom.
- The trees cannot be passed through by the controller.(Meaning that adding mesh colliders)
- The end can also make some effects such as fireworks that brings out the happy messages rather than only a text.
- The timer may change the color if there is 30 second left
- The GUI can suit in different monitor size
- The object can use 3D mesh to create(also learn it)

  →More actual and realistic

Real implementation

Booth design

- Yellow = Curtain
- Red = waiting line
- Orange = cardboard
- Blue = Kinect sensor viewable angle
- Green = board



The real image

Usage and reason

- Yellow = Curtain

→Prevent the kinect sensor to detect others who are not playing and just walk-by

- Red = Waiting line

→Prevent the kinect sensor to detect others who are not playing but waiting for play

- Orange = cardboard

→Prevent the kinect sensor to detect others who are playing the other games

Usage and reason

- Blue = Kinect sensor viewable angle

→That to indicate the usage of the things or object mentioned in the previous slide

- Green = Board

→The board is to show some brief introduction of the games inside the booth

So the player can enjoy the game

Expected difficulties when playing

- 1) People cannot know well of the controlling
- 2) It is crowded of the queue and long waiting time
- 3) People really do not know the answer of question
- 4) The Kinect cannot work well as expected as testing
- 5) The computer may overheat since placing outside

Expected solution

- 1) Some helper in the booth can help with them
- 2) There are 3 different games at all in the booth. Someone can play for others while waiting for the game
- 3) The mechanics of answering question is the time used /2 that's mean they can have a chance to answer again so as to win.
- 4) The keyboard will replace the Kinect if Kinect cannot work well
- 5) The computer placed inside the booth without direct sunlight

Actual implementation(barrier)

5 of the expected difficulties 3 of them existed

- 1) People cannot know well of the controlling
- 2) People really do not know the answer of question
- 4) The Kinect cannot work well as expected as testing

→The Kinect cannot detect the people.

→ It is believed that caused by overheat(at that time)

No others barriers have existed luckily

Immediate react

The solution made is just liked what expected.

a) Placing 3 helpers in the booth for helping someone who do not understand how to play

b) Due to overheat of kinect, the arrangement is playing with Kinect for 1 hour and stop for 15 mins for cooling down. With that 15 mins, keyboard is used for playing the game.

Further improvement(Actual)

1. The overheat of Kinect is because of the outdoor environment

→Designing the booth like indoors with fans or other cooling down equipment.

→The overheat problem may be solved

After the project

Review of the project

- Basically the finish learning of the application Unity and language is in the last day of summer holiday and a demo made.
- The demo is just a prototype of all the 2 games made until now, one is a car game and the next is this game.
- Due to some feedback of the player, the car game has deleted and start a new project on 28/9. Because of some feedback of the game, some delay have happened than the schedual

Review of the project

- Expected schedule of the game (After 28/9)

First week: Finish the setting of virtual school

Second week and third week: Finish the game system(Get ball and answer question)

The later time: Do improvement from the feedback and set the game booth as well

However, it spent one more week on the game due to the unknown reason FAAST cannot control the view angle of the game and solution figured out.(details see FPS)

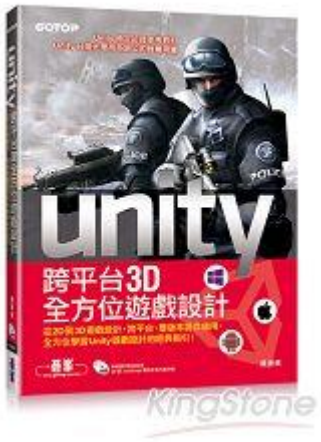
→ Unexpected problems

Feelings and reflection

The game does meet my expectation, many of the visitors play the game. In fact, something can do more just like I have mentioned in the further improvement. Learning Unity 3d making a game is really great. It can help me to have an experience of writing a game for the people to play. The time management is really a great challenge. Making a game is not an easy thing. Planning, implementing, testing and so on is just like running a business. I need to meet the taste of the players and also what kind of services I need to provide. When providing service, which is the game, I need to also consider not only the program, but also the artwork and some game balance setting such that.

Reference for learning

- This book for learning how to use Unity and JavaScript



- https://www.youtube.com/user/FlatTutorials
- A Youtube channel of Unity tutorial

End