AUGMENTED REALITY
(KINSTRIKE)

Team:

Anjali Nauhwar	2nd year (EE)	14115017
Joydeep Das	2nd year (ECE)	14116032
K. Uday Kanth Reddy	2nd year(EE)	14115055
Shubham Chowdhary	2nd year(EE)	14115115
Swapnil Lohani	2nd year(EE)	14115127
Tarun Kumar	2nd year(CSE)	14114068

Content

Acknowledgement

This project was carried out under **ELECTRONICS SECTION** of Hobbies Club, Indian Institute of Technology Roorkee. It was basically under the Entertainment projects carried out by the section.

We would like to thank our staff advisor**, Mr. Kamal Gotiyan** for his guidance and support. His immense knowledge and ideas have given us a lot of inspiration. Secondly, we would like to thank our mentor **Mehak Gupta** for her ideas and suggestions. We are thankful to **Padmanabh Pande**, secretary of Electronics Section, for his endless support, dedication and guidance.

We would also like to thank to all the people who contributed to this project, be it morally or by knowledge. Their help and support motivated us to accomplish this project.

Introduction

The Project Augmented Reality was started with the motivation of making a Counter Strike based reality game in which a player standing in the real physical world will be able to control the movement ,gestures and shooting effects of the First person player seen in Counter Strike game. The Challenge was to map a real person to the humanoid third person character in the game and to project the entire game world in 3D hologram view.

With the vision of Counter Strike game in mind, we went on to model a 3D shooting game of our own (later named KINSTRIKE) so as to explore and understand how each functionalities of the real player can be given to the humanoid character in game.

To map the real player we have used Microsoft Kinect Sensor to detect the various moments of the person and accordingly linked it to the game player by designing various animations in Unity3D for the game player. After controlling the forward ,backward ,left and right movement of the player with depth estimation by kinect the next challenge was to control the screen rotation movement and shooting of the enemy with absolute accuracy. The rotation of the screen was handled by Potentiometer device with serial interfacing through arduino and the shooting was done by taking the physical co-ordinates of the left fist of the player and mapping it to the game world coordinates of *shooting cursor*.

The Scope of Improvement in the Project lies in introduction of addition gestures to the game player like jumping, squat ,etc .Further the accuracy in shooting can also be improved by improvised techniques like having a direction vector between the left and right fist and a cost efficient designing of 3D hologram view for the game to run on.

The detailed description and Source Code analysis of our entire work is provided in the sections below.

Game development using Unity 3D

- Unity is a cross-platform game engine developed by Unity Technologies and used to develop video games for PC, consoles, mobile devices and websites. First announced only for OS X, at Apple's Worldwide Developers Conference in 2005, it has since been extended to target more than fifteen platforms. It is the default software development kit (SDK) for the Wii U. Five major versions of Unity have been released. At the 2006 WWDC trade show, Apple Inc. named Unity as the runner up for its Best Use of Mac OS X Graphics category.
- In this project we used **Unity3D version 5.3.1** for building the complete Game i.e. from making the map to adding the functionalities to the game, everything has been done on Unity3D.The Following points will give you a basic understanding of building the third person shooter game in Unity3D.
- The **Animator Controller** in Unity was used for controlling the animations of the Player as well as the Enemies. For Example animations for walking, running, shooting, standing idle, dying etc can be found in the Standard Assets Package of Unity and can be added to the model being used.
- The **Physics Engine** of Unity3D was used which enabled us to detect various objects in the map thus helping in shooting things as well as detecting collisions. The Physics Engine also produces Gravity in the whole map, but in this case it could not be used because we used **Kinect** for detecting person's gesture and hence move the player which resulted in overwriting of the Gravity function of Physics Engine thus disabling its effect. So Gravity was produced in the game by writing a C# Script for it.
- The various Avatars for the Player and the Enemy were directly taken from **the Unity Asset Store.**
- For controlling Movement of the Game Player the gestures of the user were detected by **Kinect** and were then mapped into the Unity Game Environment.
- For the various functionalities of the game the following scripts were used:

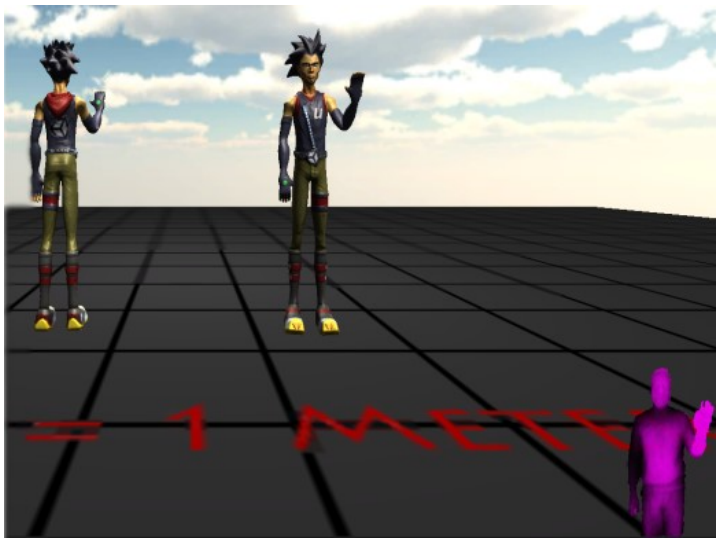
1. **Player Movement.cs(EnemyAIScript.cs)-** This Script controlled the forward, backward, left and right movement of the game player.
2. **PlayerHealth.cs-** This script maintains the information about current health of the game Player and likewise Enemy Health.cs script for enemy health.
3. **RayCasting.cs-** The shooting is controlled by this script. In order to give visual effects of shooting **Line Renderer** was used and to give a ricochet effect **Particle Systems** was used.
4. **PotScript.cs-** This script maintains the serial communication between **Arduino** and Unity. The Potentiometer reading is mapped to rotation of the screen and push button's reading triggers Gun Shooting.
5. **ScoreManager.cs**- This script maintains the score and displays it on the screen.

- Finally we also added some **GUI** to our game. A Start Screen, Instructions Page were added at the starting of the game and a Game Over/You Win Screen was added to be shown at the end of the game.

Augmented using Kinect

We have used Kinect SDK provided by RF Solutions, named Kinect with MS-SDK v1
Using this SDK we are able to have the coordinates of hands and various other joints of the body. And combining the various coordinates we can define the complex gestures such as jump, swipe, push etc.
Steps to run Kinect code in Unity:

- Install the **Kinect SDK 1.8** or Runtime 1.8.
  https://www.microsoft.com/en-in/download/details.aspx?id=40278
- Download and import **Kinect with MS-SDK**
  https://www.assetstore.unity3d.com/en/#!/content/7747
- Apply **KinectManager.cs** and Script on any object which remains alive throughout the game. Eg: MainCamera etc.



- Now we modify **KinectGestures.cs** as per requirement or create new script using these scripts.

We initially algorithm of using the movement of feet up and down to control the player's movement but it was observed to be less user friendly. Finally we coded to map the person's moves to player's moves.

- We defined the static point at distance of ~2m from kinect. Static point means if person is at that point, player will not move in the game.
- If person moves forward from that point, player will move forwards and same goes if he moves backwards, left and right.
- The position of crosshair for shooting inside game is controlled by the gun's tip and left hand(for right handed person).
- Gun will follow its own processes independently (push button etc.)

Enemy features using RAIN AI

- RAIN Unity plugin has been used to design the AI of the enemies.
- Three enemies have been placed at 3 random places in the map. We have created an AI enemy which shall patrol in a path(defined by the user) which shall automatically find an alternative path when it detects an obstacle in its way.
- When the player goes closer to the enemy the visual sensor of the enemy shall detect the player and shall start shooting at him until the player goes out of the enemy's visual sensor.
- Once the player goes out of the range of visual sensor of the enemy the enemy has another sensor known as path finder which shall start finding a path to go to the position where it has seen the player last time. and keep searching for him until it detects the player again.
- We have also build a HEALTH METER for enemy, when the player shoots him then his health start decreasing.
- We have defined ammo for the enemy which shall decrease on shooting at the player and shall enter into melee mode. In this mode the enemy's melee sensor shall detect the player when he is close to him and shall hit at the player using his gun.
- We have also defined a health meter for the enemy.If the health of the enemy is in the range of 50 to 25 where the maximum health which we have defined was 100, then the enemy shall go to a hiding spot defined by us. There can be multiple hiding spots.
- The enemy is considered dead when his health becomes zero for which we have added an animation of enemy falling down using Unity animator controller. We have added a feature that enemy disappears just after he falls down.
- All the above behaviour of the enemy has been done by creating a behaviour tree with the help of RAIN AI.

Arduino Uno for functioning of gun

The Uno is a microcontroller board based on the ATmega328P. It contains everything needed to support the microcontroller; simply connecting it to a computer with a USB cable or powers it with an AC-to-DC adapter or battery to get started.

- Arduino was used to light RED colored Led when command was given by pushbutton which is under the control of the user. When the button changes state from off to on. This is called state change detection or edge detection. We send a message to the Serial Monitor with the relevant information and we check state changes to turn on and off an LED. This is coded in such a way that LED turned on when pushbutton is pressed after a delay set by the coder.

- We have connected LED to the supply through a resistor of 10K ohms. If you disconnect the digital I/O pin from everything, the LED may blink erratically. This is because the input is "floating" - that is, not connected to either voltage or ground. It will more or less randomly return either HIGH or LOW. That's why you need a pull-down resistor in the circuit.

- A potentiometer is a simple knob that provides a variable resistance, which we can read into the Arduino board as an analog value.

- Potentiometer was used to rotate the screen of the map.

- By turning the shaft of the potentiometer, we change the amount of resistance on either side of the wiper which is connected to the center pin of the potentiometer.

- 10K potentiometer was scaled and divided into three parts.
  1st division led to right side rotation of the screen
  2nd division ohms doesn't rotate the screen
  3rd division led to left side rotation of the screen

Integration of Arduino And Unity

After the gun was designed and its Arduino code written, it was important to send its data to our game so that, that data can be used by Kinect program and other unity scripts to play the game.
So, what we did was, to basically send whatever is being sent by the buttons and the potentiometer to the serial monitor and which was then fetched up by a separate unity script. An Arduino script handled what was to be printed on the serial monitor and at the same time a unity script took whatever was printed on the serial monitor.
One major problem we faced was of the delay in actions after the button press. After googling, we came up with the idea of increasing the delay with which the data was printed on the serial monitor. This helped in removing the response delay.
Now we had two options:

1. Controlling the gun wirelessly via HC05
2. Or using it in a wired fashion

In the first method we used a HC05 module and connected it to our gun circuitry and then connect a PC to the Bluetooth. That Bluetooth Baud rate was used for the transmission of data explained earlier and the work was done. But this Bluetooth connection was a problem because of its unstable connectivity.
So, we jumped to an alternative where we simply used the USB wire of Arduino to directly connect to PC. So the 9600 baud rate was good enough for the data transmission.
Finally, in unity we mentioned the same port and baud rate as in the Arduino script and then a Serial library of C# to extract data from the serial monitor. The data received was a string so it was to be converted to integer for its use in our other scripts.

Hologram for 3D view

To create a Pepper's Ghost effect, there are two types of materials commonly used:

- The first is a transparent material that has a reflective surface. Glass would work for this, but a cheaper, lighter, and easier to use material is acrylic plexiglass. When using a transparent acrylic sheet, take a look a both sides as one side is always more reflective.
- The more reflective side should be facing the image source. Another material that can be used is a scrim or semi-transparent piece of cloth (most commonly black). Although a scrim won't reflect the image off of it, it can indeed catch an image as a movie screen might. A mirror or semi-transparent mirror would also work in order to reflect an image off it, but you would lose illusion that the object is floating mid-air.

This display system uses a technique that is often referred to as Pepper's Ghost. It was first invented by Giambattista della Porta in 1584 and has been commonly used in theatre. Pepper's Ghost uses a much simpler technique that merely reflects an image off of a surface to create an illusion of a 3D object floating in physical space.

Problems Faced

We have initially thought to detect the gun position in real word using image processing in which we could detect the red colour using the rgb values which the kinect shall provide us using its colour stream. But large amount of time(time Constraint came into play) was being consumed in processing the frame and finding out the position of red led in the real world. This was significantly decreasing our game performance.

So we decided to drop this idea and instead use the skeleton data from kinect by detecting the position of our left palm with which we would be holding the potentiometer of our gun.
As the handle was just beside the red led attached to the gun the accuracy was same as before and thus the shooting of the player was solved.

References

1.Download rain from this link
http://rivaltheory.com/rain/download/

2.Download fpscontrol project from www.fpscontrol.com

3.For downloading fpscontrol u need to first register and then verify both the emails u get to ur account and in fpscontrol.com u need to go to Mydownloads and download the project on the left side.
Then u need to import RAIN into fpscontrol project. (click on Assets>Importpackage>Custompackage>(and click on rainunitypackage, the one which u have downloaded)
After finishing the above steps succesfully, Export the package of the project which u have done till now and then import this into fpscontrol project.
www.fpscontrol.com website shall open only from Mozilla Firefox as that website has a PHP error.
4.Further useful website links are provided below:-
https://www.arduino.cc/en/tutorial/potentiometer
https://www.arduino.cc/en/Tutorial/StateChangeDetection
https://www.youtube.com/watch?v=of_oLAvWfSI
https://www.assetstore.unity3d.com/en/#!/content/7747
www.microsoft.com/en-in/download/details.aspx?id=40278