**Final Project Report**

**Team USF Bulls- AR**

**Team Members:** Akhila Palle, Harika Reddy Patlolla, Raviteja Varma Penmetsa

**Youtube Link: https://www.youtube.com/watch?v=plDw6xGnX4E**

**Description:**

The objective of our project is to create an immersive and captivating multiplayer fighting game utilizing augmented reality (AR) technology. Through this innovative approach, participants will immerse themselves in a virtual arena, engaging in real-time battles that merge the physical and digital realms to deliver an unparalleled gaming encounter. Our application name is “Augmented Warriors”. It sounds fun when we talk about the virtual character fighting games, Isn’t it? Augmented Warriors is one such application in which we can have fun enjoying the Augmented reality experience into the real world. In details let’s see the design and implementation along with the structure of the code.

In this project, photon is linked to prefabs for seamless connectivity, and player variables are synchronized throughout networked gameplay by their photon view attributes. Network movement is implemented by defining efficient logic and procedures; additional improvements include the spawning and removal of objects. When an opponent hits a prefab with a box collision, interactions are triggered, changing network variables and affecting gameplay. The functionality of the game has been thoroughly tested.

Through the use of immersive augmented reality, Augmented Warriors aims to revolutionize gaming by enabling players to see virtual fighters blend in perfectly with their actual surroundings through smartphones. One-on-one fights are made possible by multiplayer features, which up the social and competitive elements of the game. Engaging gameplay is ensured by a varied character roster and combat styles, while a responsive combat system emphasizes player skill and strategy to shape battle outcomes. Gamers can experiment with different characters and strategies, which promotes strategic and dynamic gameplay.

**Technologies Used:**

Unity, ARFoundation, ARCore, Two Android Mobile Phones (API 24 or above), Photon.

**Working:**

**Virtual-Virtual Interaction:** The virtual prefabs interact with each other on box collision. When there is a collision of two rigid bodies, their speed is taken into context and the prefab with has the lowest speed is impacted as the health of the corresponding prefab is affected.

**Virtual-Physical Interaction:** The interaction between physical object and virtual object is based on OnTrigger event between a phantom surrounding the real object and virtual object is observed. The health of the player is boosted when the physical object comes in contact with the virtual object.

**Scenes:**

There are four scenes in our project. They are the lobby scene, Player selection scene, Loading scene and the game play scene.

**Lobby scene:** The lobby scene contains the background image, text saying “Lobby/Multiplayer”, a button to start the match with text “Quick match”, a 3-D holder, a UI login in which it contains the Background Image, a login box containing the “Enter Game” button, player name text and a text input field to enter the name of the player. The UI connection status contains the UI connection status box along with a Lobby Manager.

**Loading scene:** The loading scene contains the UI Background image along with a text saying Loading.

**Player Selection scene**: In the player selection scene, one can be able to select the virtual character he/she want among the all the available virtual characters. This scene contains the UI background image, a green glow, a UI 3D holder, a UI foreground and a player selection manager. The four virtual objects are Model\_1, Model\_2, Model\_3 and Model\_4, these are there in the player switcher game object. The UI foreground contains the bottom bar i.e the select button and the back button.

**Game play scene**: In the game play scene, we need to place the fighting ground and adjust the place accordingly. It contains a “search for games” button, a UI Inform panel, adjust and place buttons in order to adjust and place the ground, a ray cast centre image and a scale slider to virtual view the size of the ground. Once we are good with the ground position, we can click on the search for games button to enter a room and start the game. The game manger, spawn manager is there along with the plane in which it contains arena (Battle ground) all the virtual characters saying spawnPosition1, spawnPosition2, spawnPosition3 and spawnPosition4.

**Scripts:**

**MovementController.cs:** The joystick prefab is used to control the movement of virtual prefabs in the real world.

**Spinner.cs:** Contains the spinning logic of every prefab and the degree of rotation is stored.

**SceneLoader.cs:** The logic of loading scenes which is implemented between every scene change.

**Singleton.cs:** Singleton is a predefined script file to effectively load different scenes.

**LobbyManager.cs:** Contains UI of different texts and buttons which are set as active or inactive in the script.

**PlayerSelectionManager:** Stores the chosen player and incorporates the logic behind choosing a player.

**PlayerSetup:** Controls which joystick should be displayed based on user and opponent.

**MultiPlayerAugmentedWarrios:** To track the chosen prefab between the scenes.

**AWGameManager:** Creates random rooms so that different prefabs can join in a created room.

**SpawnManager.cs:** The prefabs are instantiated to different locations on the arena randomly. The logic behind the instantiation is defined in this script.

**BattleScript.cs:** Contains the battle logic to make changes to health of the player. The virtual physical interaction is implemented here.

**MySynchronizationScript.cs:** The position of the prefabs is sent through networks to match the transform and rotation of the prefabs.

**ARPlacementManager.cs:** A Ray cast is casted from the centre of the camera and the first hit is chosen to place the arena from the possible plane detections.

**ARPlacementandPlaneDetectionManager:** Different UI aspects of the plane detection is controlled where the user can place and adjust the battle arena.

**ScaleController.cs:** The size of battle arena is manipulated by changing the size of XR origin of ARFoundation.

**MITM.cs:** The physical objected is detected when ‘AR Tracked Image Manager’ detects images and changes in the image events are examined to control UI pertaining to cheat activation.