Documentation

Unity Version: 2020.3.19f1

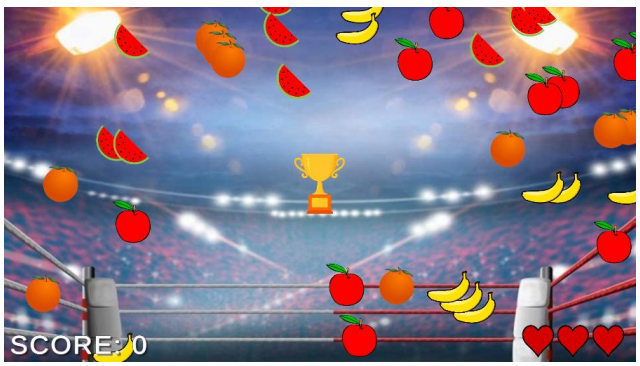
Game Developers: Kacper Grzenda (G00373427) & Oskar Grzenda (G00373428)

Project GitHub Repository: https://github.com/kacpergrzenda/Gesture-Based-UI-Development

Purpose of the Application

The game is inspired by a popular game called 'Fruit Ninja', but that game was based on using a touch screen device and we took that game concept and made it a gesture-based game using the Kinect. The purpose of this application was to get familiar using the Kinect hardware. This was our first time integrating a piece of hardware with one of our projects. The hardware we decided on using was an Xbox Kinect V2. The Kinect scans a human body using the Kinect camera and using gestures a user can control the game functionality. The main focus of the Kinect in this application is to scan and locate the user's hands, as the hands are the main tool for controlling the in-game gestures. Voice Control is also implemented into the game to allow a user to control the menu screens. The goal of the game is to protect the trophy in the middle from the fruit hitting it. On screen you can see a trophy and fruit flying towards the trophy and you have to hit the fruit before the fruits hit the trophy. The game requires you to constantly keep on moving your hands around your room and the Kinect would match your hand position to the position of the boxing gloves on screen. Below you can see an example image of what the main menu UI looks like and what Level One looks like.

Main Menu



Level One

Gestures identified as appropriate for this application

With the Kinect we had initially many different ideas for our game. The Kinect allows for a lot of different gesture-based games. We decided to make a game where it would require you to move your hands a lot and the Kinect fit this idea perfectly as the Kinect has very good body tracking features, so it was easy to set up and track our hands. We decided to create a gesture-based game that was focused on just using your hands as space can sometimes be a limiting factor when it comes to gesture based games. Sometimes people may not have the space necessary because of furniture or other things to enjoy a gesture-based game to its full potential but we thought using your hands would be a good idea as it doesn't require much space as the game can be played by standing or sitting in one spot. If there are no objects within hands reach, then you will have plenty of space to play the game. The main thing the game requires is for you to have a fast reaction time.

Voice Control Gesture recognition was implemented to the game because during the testing process we found that using the mouse to control the menu screen was an inconvenience and made the game not enjoyable as it still felt like a mouse and keyboard game. Voice Control made the user experience better as it allowed the user to focus more on their gestures and provided better control of the menu screens. We also found that using voice control recognition was better to use for the menu screens rather then their gestures as it worked better and faster allowing the user to start playing the game straight away without facing any delays in the menu screens.

Hardware used in creating the application

We used two different pieces of hardware for the development of our application. We used a Kinect and the internal microphone of the PC. The Kinect allowed for gesture-based controls using our hands and the microphone allowed for controlling the menus of the game using voice control. We used a Kinect as our gesture-based tracking device as it allowed for very precise tracking of different body parts and our focus was on gestures with our hands and the Kinect fit that role perfectly. We also decided to use voice controls in our game because when you are playing a game using the Kinect you are more than likely standing a couple metres away from your PC so using voice controls creates a much more user-friendly experience as when the user would like to control the UI of the game, they don't need to step up to the computer they can just control everything from a distance.

We had also considered using a myo armband instead of voice controls for the navigation of the UI, but we felt like voice controls would be much more suiting for this type of game as the game already requires you to move your hands a lot and we felt like moving your hands around a lot could end up triggering the myo armband when it was not intended too. Also, voice controls have a much smaller learning curve over a myo armband as people can just say simple things, they are familiar with such as 'Start Game', 'Pause Game' etc to control the UI components. With a myo armband we could have achieved the same results as voice controls, but it would have been potentially harder to get familiar with compared to voice controls.

Architecture for the solution

The Kinect is a device that is made for an Xbox so to get it working on our windows machines we had to set up the Kinect SDK. The Kinect SDK allows developers to create applications that support gesture and voice recognition using Kinect sensor technology. The Unity Pro Package Library was used to build the application as the library provides helpful resources to build Kinect-based Unity

apps. The project was coded using C# and we used Visual Studio code as our IDE. We also used Unity as our Game Engine for the development of our game.

SOLID design principles are strictly followed in the architecture of the application, as the design of the application is large and keeping the code easy to understand and well maintained was a priority. As change was a key factor in the application because once a test was complete and a better solution was found for the application, the change was made. The SOLID principles helped with making changes as the code was clear and changes did not impact other areas of the code.

Conclusions & Recommendations

After taking on this project we have learned that gesture controls can add a whole new aspect/experience to a game. We turned a game that was meant to be played on a mobile device without much physical movement to something that will require you to move a lot more and after a long time of playing it will almost become an exercise.

We found that combing two gestures to work together can make a massive difference, as in this case we combined the Kinect and Voice Control, which worked way better then we imagined. It made the game more user friendly but not only that made the testing process of the game easier as it eliminated the constant use of the mouse for the menu screen. Adding the Voice control gesture allowed for more focus on the actual gameplay and use of Kinect gestures.

Making a 'Fruit Punch' game was definitely a good way of getting familiar with using a Kinect and creating a gesture based game but if we were to take this project on again, we would go with a different style of game for example a game that involves the whole body as we really enjoyed the aspect of turning a sedentary game into something that is quite active and can become a form of exercise for someone while still playing a game they enjoy. But we decided to make a game that focuses on the persons hands as we felt it would be more suiting for most people who may not have the sufficient space around them to play a game that involves their whole body and moving around the room but next time creating a whole-body game is something we would be interested in giving a go.

References

Main Menu

https://www.youtube.com/watch?v=zc8ac_qUXQY

Game Over Menu

https://www.youtube.com/watch?v=ZfRbuOCAeE8

Music

https://www.youtube.com/watch?v=DzFXGsRvSwA

Kinect SDK

https://www.microsoft.com/en-us/download/details.aspx?id=44561

Kinect Set Up with Unity

https://www.youtube.com/watch?v=aHGlLxh6a88