Gesture based UI Project

By Cormac Raftery (G00348802)

Introduction

This document looks at the general-purpose documentation relating to the project. This project was completed as part of the Gesture Based UI Development module in the 4th year software development course. This project carries 75% of the overall grade for this module.

For this project, I developed a gesture-controlled adaption of the original “Tic Tac Toe” game as well as the origin “Connect 4” game. The rules of these games are simple, in Tic Tac Toe the players choose the first shape to play, then where to play it and alternate turns. The game is won by connecting 3 shapes in straight line; and drawn if the grid is filled up. In connect 4 the players choose which color to play first, the where to play it and alternate turns. The game is won by connecting 4 colours in a straight line; and drawn if the grid is filled up.

Research

Before considering what gesture-based project we wanted to create, a research phase was undertaken. Areas that were looked at included other gesture-based projects, what hardware was available, and what sort of projects would work well with gesture-based technology. After careful consideration, we narrowed down a list of ideas and decided to stick with a UWP project.

We looked at viable options to incorporate gesture technology, with the Myo Armband and speech recognition being available and easy to use. This was followed closely by what each technology could do to enhance the project. We looked at certain types of games that could be created using the technology that was decided upon. This swayed us into choosing classical games as we were already familiar with them.

We noticed that there were no easy to find classical games with speech recognition done online and figured we could fill that gap. Once we decided what games to do each and the hardware and software had been chosen, we looked at the purpose of the application. We agreed to keep coding in the main pages to a minimal as to not obstruct each other as working through someone else’s code is not an easy task.

Original Project

The original plan for the project was to make both of these games compatible with the myo armband but unfortunately due to the coronavirus crisis the myo armbands were made unavailable and the project had to be adapted to use whatever technology we had available, which was in my case speech recognition.

Development Process

Upon receiving the project, I knew that I was going to make a game of some type to which my partner at the time agreed with. We eventually decided that the best course of action was to try to recreate classic games and combine them into a single project. I choose to do tic tac toe and connect 4. I started with tic tac toe as I thought that would be the simplest game out of the two. I decided to get the game working with a keyboard before adding other gesture-based technologies. After some time, I successfully recreated the game and connect 4 used a lot of similar methods. Then I decided to add the voice control and had great difficulty converting the game as I could not directly send the button object through the methods and could not simply target the xaml code as it did not exist. I finally figured out that even though I could not send the object itself through the methods I could create a selected object in the position of where I wanted to play it on the grid and add the object. As I was passing a grid reference when I designed the game to work with mouse clicks, I had to make a copy of the grid itself into a Grid variable for speech recognition.

Test Plan

Features to be tested

Tic Tac Toe					
Feature	Test Case	Pass Result	Failed Result	Result	Comments
Alternate turns	1	After each turn the shape is changed	Same color is played	pass	
Draw	2	Draw screen appears after grid is filled	Game over is not recognized	pass	
Win	3	After 3 in a row is matched game ends	Game over is not recognized	pass	
Start TicTacToe	4	Game loads upon selected	Game does not load	pass	
Home Page	5	Home page loads when app is started	Home page does not load correctly	pass	
Shape	6	First shape played matches selection	Shape is incorrect	pass	
Grid selection	7	Shape is placed when user says name of grid or selects grid	Shape does not place in selected grid	pass	
Already selected	8	Shape is not placed and turns do not switch when a grid has already been picked but is selected again	Shape is changed	fail	Pass for mouse, fail for speech
New game	9	Grid is cleared and a new game is started	Grid is not cleared, or a new game does not start	pass	

Connect 4					
Feature	Test Case	Pass Result	Failed Result	Result	Comments
Alternate turns	1	After each turn the color is changed	Same color is played	pass	
Draw	2	Draw screen appears after grid is filled	Game over is not recognized	pass	
Win	3	After 4 in a row is matched game ends	Game over is not recognized	pass	
Start connect 4	4	Game loads upon selected	Game does not load	pass	
Home page	5	Home page loads when app is started	Home page does not load correctly	pass	
Color	6	First color played matches selection	Color is incorrect	pass	
Grid selection	7	Color is placed on lowest available row in column when user says name of grid or selects grid	Color does not place in lowest available row in column	pass	
Already selected	8	Color is not placed and turns do not switch when column is full and is selected	Color is changed	pass	
New game	9	Grid is cleared and a new game is started	Grid is not cleared, or a new game does not start	pass	

Gesture Implementation

To import speech recognition into a UWP you must first give the application access to your microphone which is done by double clicking the Package.appxmanifest file in the solution explorer, clicking capabilities and checking off the microphone capability in the list.

Then you must import Windows.Media.SpeechRecognition into any cs file using speech recognition. From there you can create a speechRecognizer variable and make it wait asynchronously until you speak. To listen for certain keywords I gave my speechRecognizer a constraint file in which I listed out all my keywords as commands and any way the user might try to say them.

System Architecture

The system architecture is a simplistic one for this project. This was intentional as it allowed for me to focus on the gesture-based technology.

The hardware (microphone) is an external device which is tied into the software. The microphone uses windows speech recognizer to allow the Visual Studio to get a handle on the microphones input. This is all controlled within the CS scripts.

Hardware Justification

The main intention was to fully immerse the project fully into gesture-based controls, with no keyboard input. However, I decided to keep the mouse and keyboard input instead of replacing it as it did not affect the speech recognition. I feel as both games are two player speech recognition works well with the project rather than having to both use the same mouse/keyboard to add their input. I left the start game function and the first player select function to mouse only as they are only need once per game and not required to be used by both players.

Game Walkthrough

When the game is initially loaded, the user is met with a home screen. On this screen there are three buttons and a command bar. Two of the buttons on the home screen start a new game and one bring you to instructions on how to play. You can also start either game or go home from the command bar.

Upon selecting tic tac toe, you are produced with the option to choose the first player's shape and then the grid appears. The game of tic tac toe is now underway, and the first player may place their piece by saying any of the command words such as "top left" or click the grid they wish to place their piece. When the game is either won or drawn the user is then presented with a game over screen and can choose to start a new game from the command bar if they wish.

Upon selecting connect 4 you are produced with the option to choose the first player's color and then the grid appears. The game of connect 4 is now underway and the first player may place their piece by saying any of the columns from 0-6 or click the column they wish to place their piece. When the game is either won or drawn the user is then presented with a game over screen and can choose to start a new game from the command bar if they wish.

Learning Outcomes and Conclusion

Although the I was quite happy with the outcome of this project, there are certain aspects that would be reconsidered. If this project was to be redone, more time would be put into considering the inclusion of a multitude of gesture-based hardware. For example, the games would be compatible with the myo armband technology.

As this was my first gesture based project I have learnt that transferring keyboard input to speech input can be rather difficult and if I was to redo the project I would build the games with speech recognition from the start rather than adding it in after the game has already been completed. I have refined a magnitude of skill, such as version control, proper planning and working with new and innovative technologies, during the development of this project.

As the original concept was to have a collection of classical games, I feel that to have only 2 games is not considered as a collection and I would like to have added more such as battleship or a memory game.

I have also learned that when you do not give a user very specific input, they seem to get very creative so you should ensure that you accept multiple responses. Longer sentences seem to be more accurate when using the speech recognizer.