**3DG Arcade**

by Evan Arroyo and Joshua Sims

Our project, titled “3DG Arcade” is a suite of 3D, gesture controlled arcade games. We chose this project because we wanted to create games which are entertaining and intriguingly original. Since the inception of this project, we have developed a 3D, gesture controlled adaptation of the class arcade game called Pong. Our adaptation is doubtlessly entertaining, the gameplay and mechanics are replicative of the exceptionally successful classic Pong, and the 3D aesthetics and gestural control distinguishes it from its older, traditional counterpart – it provokes discovery as well as induces nostalgia. The same will be true of other games featured in the suite.

multimedia (scatter it around as needed)

data flow diagram

interaction flow diagram

UML diagram

screenshots of the current system

MOSCoW analysis

Our suite of games is built with Unity and C# and is designed to be played on a PC with the Kinect for Xbox One. We could have used Unreal Engine 4 or JavaScript or Boo. We could have

We used BinaryFormatter for serialization (persistence) instead of JSON…

We used ovoids rather than rectangular prisms…

We save on each goal

Throughout our process, we used GitHub for version control and project management.

We could have created entirely original games…

use case…

Someone wants to play classic arcade games, but with an interesting, new twist…

To run 3DG Arcade, the user must first download the executable named “3dg-arcade.exe” from our GitHub repository @ <https://github.com/J-o-s-h-S-i-m-s/3D-Gesture-Controlled-Arcade-Games> then the user must run the executable. The 3DG Arcade main menu loads. The user can then navigate to particular game by clicking on an appropriately titled button. Then the appropriate game main menu loads. From there the user chooses players, clicks to play the game, and can click to view statistic or click to go back to the 3DG Arcade main menu. From the game, the player can choose to quit and return to the game main menu or just play. From the

The project works well

Our navigation menus work well

Pong works – just missing settings and..?

We did not use TDD, unit tests (or integration tests) because we wanted to focus on developing features. It is the definitely the better practice to use TDD, but we chose to exclude TDD, unit tests, and integration tests from our time budget.

We encountered problems…

Thankfully, the absence of unit tests and integration tests was not related to any bugs.

The ball getting caught along the barrier

The ball incessantly ricocheting

The ball not bouncing off the paddles at angles

Paddles composition - not contained within barrier, but also must hit the ball and let the ball bounce

Ball would shoot out of the barriers so rebuilt the arena

Implementing the Kinect

The courses that helped us make informed decisions and fix problems were…

brief on statistics…

We collected data regarding the player’s gameplay.

People usually use that data to showcase their skill, set records, use as a guide for improvement, matchmaking, or awards.

non-network difficulty…

tracking the statistics to award the player whenever a certain statistic reaches a certain value, record records, advise the player based on statistics, generate an overall skill score.

network difficulty…

It would not be difficult to screenshot the statistics and share them across the network as an image file or as a text file, but unless some kind of certificate was incorporated into that data, there would be a shortcoming in its plausible validity. If the statistics were shared in-game such that the game is a distributed application then the statistics would have a far greater credibility, but the networking aspect of distributed applications is difficult to manage because networking introduces another variable of difficulty.

detailed statistics…

we gathered statistics on…

the source is the player’s interaction with the game.

list user reqs…

the project followed the timeline well…not well…

We would have done differently…

If this was a work atmosphere instead of a school atmosphere, we would have incorporated the unit tests.

We plan to…

Further develop Pong – refactor, implement unit tests, include split screen, include settings, push to start the match, are you sure you want to quit

develop another game – galaga, dig dug…