Natural User Interface and Virtual Reality Integration in Video Games

Last Revised: February 1, 2015

Alexandre Wimmers adwimmers@ucdavis.edu
Mario Yepez myepez@ucdavis.edu
Timothy Tong tktong@ucdavis.edu
Valerie Gadjali vgadjali@ucdavis.edu

1 Summary

Since its birth, video games have continuously evolved — improving graphics, inventing new genres, integrating new mediums, etc. Starting as arcade video games, it later evolved to be consumer-owned systems starting with the Atari 2600. As system performance improves, a greater variety of mediums emerged. Games can now be played on the computer, on a console, and through the internet.The newest medium that has emerged recently is both the natural user interface and virtual reality.

Natural user interface is the ability to use an application without the need for any controller. In this case, the players own body or parts of the body becomes the controller. The concept of natural user interface first appeared in 1990, but the first popular usage for video games appeared in 2010 with the release of the Microsoft Kinect. Traditional keyboards and mice can only interact with a monitor or any visual interface in a two-dimensional sense. However, with a natural user interface it becomes possible to interact in three dimensions.

Virtual reality is an immersive experience within a computer-simulated environment. First conceptualized in 1930, but only recently has it become a popular with the Oculus Rift Development Kit 1. It provides the user a complete immersion and a new perspective into the virtual world. Whereas a monitor restricts a user to only view in two dimensions, virtual reality allows the user to see the world in three dimensions.

The combination of both natural user interface and virtual reality is an idea that has only appeared within the last 1-2 years. However, there are still very few video games that utilize these new mediums. The goal of this project is to take an existing genre and develop a video game of that genre integrating both natural user interface and virtual reality.

The genre of choice is Castle Defense. The primary goal of the game is to protect the castle and survive. Waves of monsters will attempt to breach the castle and the player will have to either defeat the entire wave or hold off for a set amount of time. One of the classic castle defense games — Defend Your Castle — is a flash-based browser game limited to only two dimensions and mouse-clicking.

2 Stakeholders

The stakeholders are Alexandre Wimmers, Mario Yepez, Timothy Tong, Valerie Gadjali, Aveek Das, Dr. Prasant Mohapatra, and Dr. Xin Liu.

Course instructors and assistants – Aveek Das, Dr. Prasant Mohapatra, and Dr. Xin Liu – shall act as advisors overseeing project progress and providing support.

Because this is a student proposed and self-funded[1] project under course 193AB in the Computer Science Department at the University of California, Davis, the decision making shall reside with the students of this project – Alexandre Wimmers, Mario Yepez, Timothy Tong, and Valerie Gadjali. A single signature from any member of the team will signify client approval of this document.

3 External Ecosystem

3.1 Hardware/Software Requirements

The hardware requirements are a minimum of two computers, two Leap Motions, and two Oculus Rifts. This project implements multiplayer capabilities, therefore a minimum of two of each devices is required.

- The Leap Motion is a device specialized in receiving hand gestures at 1/100 mm precision and at over 200 frames per second. In general, It was chosen over the Kinect 2 because of its specialization, cost effectiveness, and integration with both the Oculus Rift and Unity game engine.
- The Oculus Rift is the most well-known and popular virtual reality device providing users with a deep immersive experience. With the release of the Oculus Rift Development Kit 2, there is positional tracking and improved resolution. The Oculus Rift is also supported on the Unity game engine.

The software requirements are Unity, Blender, Oculus SDK, and Leap Motion SDK.

- Unity is a freemium[2] game engine using either UnityScript[3] or C#. For this project's purposes, the free version will suffice since it provides support for Leap Motion and Oculus Rift. It is available for development on OSX and Windows, but deployable on all major systems as well as browsers.
- Blender is a free open source 3D graphics software. It will be used to create visual effects and models. Blender is available on all major systems and resulting models are exportable to Unity game engine.
- Oculus SDK is the source development kit for the Oculus Rift. Used for Oculus Rift customizations and integration with the Unity game engine. Available on all major systems.

[1] Self-funded as of January 22, 2015.
[2] Free and subscription based version available.
[3] A derivation from Javascript.

- Leap Motion SDK is the source development kit for the Leap Motion device. Used for customizing the Leap Motion (custom hand gestures) and integration with the Unity game engine. Available on all major systems.

3.2 Other Concerned Systems

The purpose of this project is to demonstrate the idea of a true three dimensional real-time castle defense game. Therefore, the first iteration will be exclusively in the Unity environment and will not be concerned with other systems.

3.3 External Critical Data

The project will be built entirely from scratch. There is no existing critical data since there does not exists a castle defense game that utilizes the Leap Motion, Oculus Rift, and Unity game engine.

3.4 User Interface Guidelines

The goal is to have input purely from the Leap Motion and Oculus Rift. The Leap Motion records hand movements and gestures while the Oculus Rift provides camera rotations all of which translates into having an effect on the virtual world.

4 Functional Requirements

4.1 User Interface Finalization

User interface specifications are subject to change based on prototyping. A proof of concept for specific interface controls, usability, and organization will be necessary in determining the final details. This project will be following Agile development and iterative development methodologies for adaptive planning.

4.2 Detailed Specifications

4.2.1 Interface

1. Primary:
   (a) Display virtual world through Oculus Rift.
   (b) Camera rotates and shifts with Oculus Rift.
   (c) User Controls:
      - Camera Zoom.
      - In-Game Pause Menu.
      - Menu Interactions.
      - Attack Methods.
2. Secondary:
   (a) Unit Spawn.
   (b) Base Repair.

4.2.2 Game

1. Primary:
   (a) Multiplayer Connectivity.
   (b) Different Modes:
      - Single Player Survival.
2. Secondary:
   (a) Multiplayer Survival.
   (b) Multiplayer Battle.

4.2.3 Low Priority

- Improved 3D Models.
- Expanded HUD.
- Sound.

4.3 Functional Testing

Correctness of program will be verified through automated testing and QA testing. Build pipeline will consist of unit testing, integration testing, and acceptance testing.

5 Non-Functional Requirements

5.1 Security

There are no runtime security requirements for this project at the moment.

5.2 Fault Tolerance

Main area of concerns are incomparability issues in versions between Leap Motion, Oculus Rift, and Unity game engine. Because the goal of the project is to demonstrate an idea, for simplicity and development purposes, the versions of all technology will be restricted to the following until further notice:

Leap Motion SDK	2.2.2.24469
Oculus Rift SDK	0.4.4 Beta
Unity	4.6.1

Another issue may be software and hardware crashes. This is not a critical issue since there is no need for data recording. Therefore, the magnitude of this issue is the same as every other application on a computer.

5.3 Performance Requirements

For a majority of games, frames-per-second (FPS) is a typically benchmark used to measure the performance of a game. Performance is typically tiered as follows:

Frames-Per-Second	Performance
< 30	Limited
$30 - 40$	Average
$40 - 60$	Good
> 60	Optimal

The Oculus Rift, however, supports up to 70 FPS, and it is strongly recommended for developers to ensure that their games run at such a performance in order to prevent discomfort from experiencing a choppy virtual world. As such, 70 FPS will be our performance goal.

5.4 Scalability/Throughput Requirements

For simplicity in demonstrating true real-time gameplay, the project will utilize the minimum number of devices that communicate directly via LAN.

5.5 Non-Functional Testing

All non-functional requirements, as specified above, can be verified by user run-through of the game.

6 Process Requirements

6.1 What Must be Done When?

By the end of March, a simple prototype of the castle defense video game will be completed. It will provide the basic gameplay of single player survival. After this, the second prototype is to be completed by mid-June, which will be more finely tuned to make game more aesthetically appealing and contain all the different modes.

6.1.1 What will be Delivered, and to Whom?

A fully functional Unity game with multiplayer capabilities will be delivered to the advisors. The source code and graphics will be retained by the students of the team.

6.1.2 In What Form will it be Delivered?

A folder with the Unity executable and supporting files necessary will be delivered to the advisors.

6.2 Who Will Sign Off When it is Done?

A representative of the team will sign off.

7 Summary

For our initial prototype we aim to produce a Castle Defense game with the Unity game engine, Oculus Rift, and Leap Motion. With these technologies we will provide an immersive and natural experience to users through the use of motion controls and virtual reality. We will have a variety of game modes to choose from which range from Single Player Survival to Cooperative and Competitive Multiplayer Modes. For our first iteration we will provide a prototype that implements the controls and renders a virtual world for the player and provides the basic gameplay of the single player survival mode. We will continue to make improvements to the system by adding more game modes, refining the controls, making improvements to the interface, and making the game more aesthetically pleasing.

8 Signatures

Team Representative

_____

Name (Printed)

_____

Signature

_____

Date