

# VIRTUAL REALITY SPACE DEFENDER

TEAM WINDSONG

MICHELLE DOBBS

EVAN BERNSTEIN

## PROBLEM & SOLUTION

Virtual Reality (VR) is a rapidly emerging platform for computing, but most people don't know how it works, what it's for, or what its potential applications are. We believe it is crucial to spread the word about VR so that, when it becomes ubiquitous, people understand it and do not fear it. To this end, we have set about creating a game that clearly demonstrates some of the fundamental concepts of virtual reality and showcases a few of the key features at the core of the technology. The game involves two players, each with a smartphone, competing against each other in an asymmetric fashion. One player will be immersed in a VR experience by inserting their device into a Google Cardboard apparatus and attaching it to their head. The other player will not experience virtual reality, but rather will affect the VR player's world. This "God Player" will see a bird's-eye view of the game and can see the VR player's movements and actions represented on the screen. The VR player's experience will be affected directly by the God Player's actions, and they will have to immediately adapt to these changes.

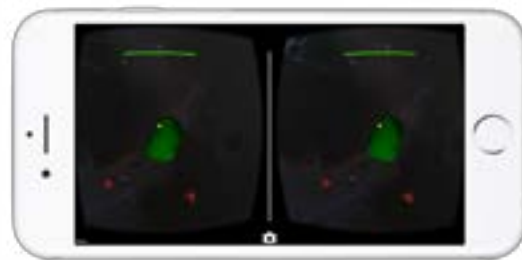


Figure 1

VR PLAYER



Figure 2

GOD PLAYER

## APPLICATION DESIGN

VR Space Defender is fundamentally an asymmetric application—that is, the experience of the God Player is much different than that of the VR Player. To keep both players in the same world, however, and to take advantage of the first-party networking APIs Unity offers, both players exist in the same scene in Unity, but have different cameras, view modes (2D vs. 3D), and interfaces. The VR Player has a split-camera view, provided by the Google Cardboard API, to produce an immersive VR experience when wearing a Cardboard headset. When the VR Player gazes at an asteroid, it turns green and a cursor is shown. Clicking the cardboard trigger at this point destroys the asteroid and generates an explosion sound. The VR Player can also see their health, displayed by a shrinking green arc at the top of their display, and is told the direction of incoming asteroids by red arrows which appear as needed (see figure 1). The God Player's interface also includes a display of the VR Player's health, as well as a symbol which indicates the direction the VR Player is facing as well as a ring which denotes the spawning location for new asteroids (see figure 2). When the God Player taps on the screen, an asteroid spawns at the corresponding location along the green circle and begins hurling towards the VR Player. The game ends when the VR Player's health reaches 0 (it is originally 10) or time runs out (120 seconds are allotted).

## EXPANSION OPPORTUNITIES

Although VR Space Defender is feature-complete with respect to a minimum viable product for demonstration and educational purposes, there are many areas where its functionality can be increased and extras can be added. As is, the game includes one set of graphics for skyboxes, asteroids, player tokens, and health indicators. With more time, appearance-customization could be implemented to allow the user, perhaps in a separate menu, to change the look of the game to their liking. Different kinds of asteroids could also be spawned at random, or in a user-controlled way to add some variation to the gameplay. Other objects besides asteroids could also be spawned, for instance health-increasing items or time-warping power-ups. Beyond the cosmetic, the game could also be expanded upon to allow for networked play between more than two players, i.e. multiple defenders and/or multiple attackers. Although the game is currently designed for a local connection, it is conceivable that it could be implemented for WAN play as well.

## TECHNICAL IMPLEMENTATION

VR Space Defender takes advantage of Unity's NetworkManager system to send messages between the client (VR Player) and server (God Player). When the God Player creates an asteroid, a prefab with a network identity is spawned on the server and assigned client authority. The movement of the asteroid is predetermined based on its spawn location and updated locally on each device (so position information is not sent over the network after the initial spawn). Messages are sent from client to server when an asteroid is destroyed by the VR Player. Otherwise, asteroids are assumed to have hit and each device decrements the health accordingly. The game begins when the VR Player joins the session, so the client keeps a game timer and sends the server a message when time has expired. The client also sends updates to the server about its current direction. Client/Server communication is handled by [Command] and [ClientRpc] functions, and if-else statements are used frequently to implement the asymmetric player experience.

## SUMMARY

As a tool for exposing end-users to the nature and possibilities of virtual reality, we believe that VR Space Defender could be a valuable and useful platform for hands-on exploration. Although the gameplay is rather limited (as necessitated by the constraints of a 10 week undergraduate course), the reactions we have observed from friends who have played has been overwhelmingly positive. Using games as a way to introduce users to new platforms is an idea that we hope the pioneers of VR will whole-heartedly embrace.

## ACKNOWLEDGMENTS

This project was completed as part of CS 234/334 Mobile Computing (Winter 2016), taught by Prof. Andrew A Chien, with TA support by Yun Li and Yan Liu.