# VR First Person Shooter Game in Minecraft World

Liuming Zhao
Stanford University
Electrical Engineering
lxz299@stanford.edu

## Abstract

*In this project, I designed first person shooter game with Minecraft background. The player in the game has to shoot in total of 5 monsters in order to win the game. In the meantime, monsters can also shoot to the player with the bullets. Each monster will automatically choose one direction to move and wait for 3 seconds and make the shot during the waiting time. The player can withstand at maximum 20 shots from monsters. If the number of shots exceed 20, then the player will lose the game.*

## 1. Introduction

My first person shooter game will utilize homework 6 as the starter code and third party library to load the gun geometry and texture [3,4]. The language that I used to build this game is THREE.js and used homework 6 as the starter code to build this game. I used Minecraft as the game background. The player can either use light house or buttons to make transitions during the game. If the player prefer to use buttons to control the transitions, "B" button should be pressed so that the game can enter into the mode where the user can move forward and backward by pressing "right", "left""up" and "down" buttons on the keyboard. If the user prefer to use light house to track the positions, then the player can press "I" button on the key board to activate this mode. If the player choose to use playhouse to track the transitions, then the user should directly face to the base station. Since the four photodiodes cannot face back to the base station, the pose tracking performed by light house only works when the player faces to the base station. By default, the game will start with mode which the player use the button to control the transitions. In addition, I also utilized IMU to track the orientation angle of the player and the direction of the shot gun.

The player can press "return" button to make the shot and a black sphere will be placed represent the bullet. The bullet will disappear if it is hit by the Minecraft wall or the monster. The bullet will also disappear if the distance of the bullet is too far from the shooter (player or the monster). In addition, I also make the y-axis (height) position for the both of the player and monster align with the height of the Minecraft block. The figure below shows the monster faced used to build up monster block in the game:

Figure 1. Monster's face [2]

In total, there are 5 monsters moving around inside the Minecraft block. Each monster will randomly pick up a direction each time and quickly make the move. After the monster has made the movement, it will wait for few seconds and then fire the bullet (block ball) at the player. The player should choose to avoid the gun shot by also make the movement either through the "up" or "down" key to move forward or backward or use the light house to physically make transitions (depending on which mode you are in). If the user get hit by the bullet, a flash of red will be displayed indicating that the player has been shot by the monster. In total, the user can withstand 20 gunshots from monsters. If the number of gunshots exceed 20 times, then the player will be dead and the screen will turn into total red and the game terminated. The figure below shows the starting scene of the game:

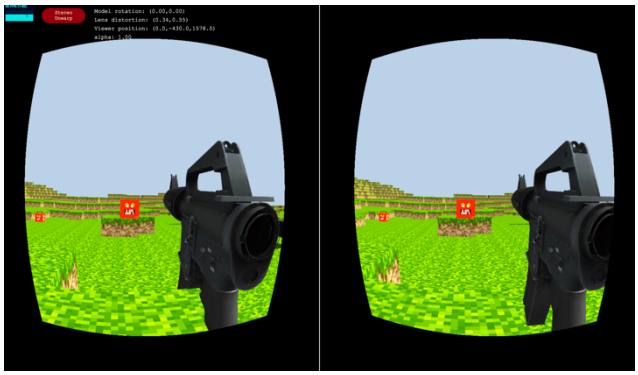

Figure 2. Starting scene

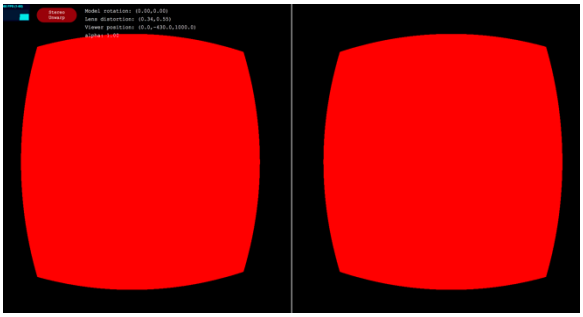

Figure 3. The scene when player lose the game

## 2. Hardware and software implementation

In this section, I will briefly introduce hardware and software implementations of this game. For the hardware devices, I mainly used head mounted display in the homework and VRduino for for pose tracking. The figure below shows the hardware setup for the game:


Figure 3. Hardware setup

### 2.1. Minecraft background setup

I built the Minecraft background mainly according to the example codes in the THREE.js website in which they demo the Minecraft example using html [1]. Basically, I first randomly generated an array of floats according to the size of the whole Minecraft world width and height. Each number in the array represents the height of each Minecraft building block (cube geometry with the texture loaded from the image on each surface). Thus, I can generate a 2D Minecraft arrays with different heights. The figure below shows an example of Minecraft world illustration which is implemented according to [1]:

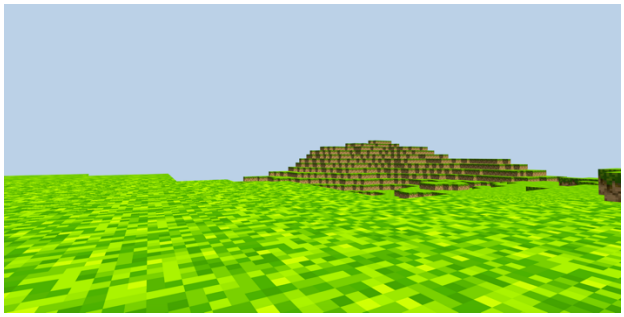

Figure 4. Minecraft world illustration [1]

In my won implementations, I first generated an array of random numbers representing the height of each Minecraft building block. Then, I export the values of this array and hardcode them in the StandardRender.js file such that each time and setup of Minecraft world is the same. I also compare the user's height with the building block's height to make sure that the user's height is always align with the building block. So does the monster. This part of implementation is also shown in StandardRender.js file.

### 2.2. Shotgun geometry and texture

I created Shotgun mesh using the MTLloader.js and OBJloader.js from [3]. The creation of gunshot mesh is also placed in StandardRender.js file. The geometry and material information are loaded using MTLloader() and OBJloader functions. The texture is also from [3] to create the mesh for the shotgun. I attached the position of this shotgun to the camera such that the shotgun mesh is static with respect to the camera position.

### 2.3. Object Movement and Collision

After the player hits "return" button to make the shot, the direction of bullet will be calculated by the quaternion information provided by IMU. I simply used applyQuaternion() function to directly apply quaternion to the (0, 0, -1) vector to get the current direction of the camera. Then, I converted the direction vector into spherical representation to get the angles with respect to x-axis and y-axis. For each frame, I will update new 3D position of the bullet along this direction.

The collision between the monster and bullet will be calculated by L2 distance in their 3D world positions. If the L2 distance between the bullet and the target is smaller than certain threshold, then I will remove both of the bullet and the monster together from the scene so that they will be disappear. This part of the code can also be found in the StandardRender.js file.

A more detailed list of any third party libraries that I use in my project is described in the readme file of my

code. In general, I mainly used js utility libraries from THREE.js official website [1] to construct Minecraft background and the model file of the shotgun from another first person game [3].

## 3. Future work

In the future, I plan to add more functionality to this first person shooter game. One possible and interesting plan is to add more Minecraft building blocks instead of just the grass and walls shown in figure 4. Due to time constraint, I didn't add the healthy bar indicating the remaining shot that the player can withstand from the monsters. This feature can also be added in the future.

## References


[1] https://github.com/timoxley/threejs/tree/master/examples
[2] https://fineartamerica.com/featured/funny-monster-face-gaspar-avila.html
[3] https://github.com/davidinfante/First-Person-Shooter-JS
[4] https://stanford.edu/class/ee267/syllabus.html