A FIRST-PERSON SHOOTER GAME DESIGNED TO EDUCATE AND AID THE PLAYER MOVEMENT IMPLEMENTATION

Chunhei Zhu[1], Yujia Zhang[2]

[1]Beckman High School, 3588 Bryan Ave, Irvine, CA 92602
[2]University of California Irvine, Irvine, CA 92697

ABSTRACT

The issue of finding a clean and simple player movement implementation that the general public will find intuitive and easy to use has been tackled over the years in various ways. With FPS (first-person shooter) games, the need for a simple and fun style of movement is monumentally crucial, as that will be a core aspect of the gameplay [4]. To address this issue, an FPS game was created with the ability to maintain momentum while crouching with intention of providing a smoother and more intuitive gaming experience for players. This movement implementation was tested by having participants play the game for a sufficient amount of time, then asking the participants to rate the experience of movement in the game and the overall enjoyment of playing the game. The results indicate that the implemented movement would be well-received by the general public, as the vast majority of the participants viewed the new form of movement as a welcome feature based on the optional feedback and the quantitative ratings. However, the other aspects of the gameplay were not as polished and therefore lowered the overall enjoyment of the game for the participants, particularly the shooting in the game that does not yet have proper audio or visual cues to let the player know that the weapon has been fired.

KEYWORDS

FPS Game, Player Movement, Unity, Artificial Intelligence

1. INTRODUCTION

First-person shooter games are a genre of shooter game generally based around guns or ranged weapons, in which users would control the in-game character in a three-dimensional space (3-D) [5]. Many FPS games' goals are similar; multiplayer games usually either involve different teams fighting against each other in a player versus player setting (PvP) or players working together to defeat a common enemy in a player versus environment setting (PvE), whereas single player games tend to involve elements of exploration and adventure [6] [7]. The history of first-person shooter games started in 1973 when the first FPS titled “Maze War” was developed. However, it is believed that the concept of FPS games was solidified in 1992 with the release of the game “Wolfenstein 3D” [10]. First-person shooters rely on the first-person point of view, which is through the eyes of the character in the game. A benefit to having a game be an FPS game is that it can be more realistic and/or immersive since the game is in a 3-D environment. People can enjoy the first-person perspective and feel like they are in the game by controlling the character in the game from this first-person point of view, which would not be possible if they were playing a game from a third-person perspective. Most first-person shooter games have some realistic factors, and an example is walking slower after getting damaged by an enemy. First-person shooter games are significant for their massive popularity and their large impact on the gaming

David C. Wyld et al. (Eds): MLSC, ITCSS, ACSTY, SOFE, NATP, BDAB - 2023
pp. 37-48, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130203

industryas a whole. FPS games are relatively simple to get into since almost all FPS games have the same keys to control the character and move around. Furthermore, many multiplayer games have a competitive aspect, which may draw some players in. The aforementioned immersion is also what some players like to experience when playing FPS games [8].

Many FPS games currently exist, and they have been a popular genre of game. An example of an FPS game is Valorant, which is an online multiplayer game that pits a team of five against another team of five. However, some potential issues exist within these games, specifically with the movement of the player. In Valorant, the player slows down while crouching, which is quite common. However, while this game development decision makes sense from a logical standpoint and keeps a tradition that was passed on from previous FPS games, such an implementation may come off as jarring and limiting to some players. Some players prefer to maintain their momentum in the game whether they run normally or crouch, which can be frustrating and disappointing when a game is not designed to do that. Another issue that Valorant has is that as an online multiplayer game, a poor connection could result in an unenjoyable experience. Movements and shots may not register, and what the player sees may not be up to date with what is actually occurring in the game.

Therefore, only those with stable and fast internet connections will be able to enjoy the smoothest gamingexperience. does not suffer from the issue. The final drawback of this game also has to do with the nature of multiplayer games, which is the toxic environment that players can potentially be placed in when interacting with both voice chat and typed chat. Disgruntled players may insult the opposing team or eventheir own teammates in frustration due to losing the game. Those who play the game with this type of player will generally not be able to enjoy the game as much, as encountering such players creates a stressful and uncomfortable situation that may leave others in a bad mood long after finishing the game.

An FPS game created in Unity was the tool used to address the issue of player movement in other video games. In the game, the player can move around a map that resembles a city with houses and buildings and shoot enemies with a gun. The player can jump on top of the buildings, and the player can destroy the enemies once enough the enemy takes enough shots from the player. The enemies will navigate toward the player and attempt to walk into the player. If the player stays in contact with an enemy for long enough, the player loses and has the option to either restart or quit the game. One important implementation in the game that is intended to enhance the player's movement is the maintaining of player momentum, whether the player is running or crouching [9]. As many other games slow down the player while the player crouches, this game takes a different approach in hope that the player has a more pleasant experience operating the character. The player also has a high movement speed, which can allow for more control. As this is a single-player game that does not require any internet whatsoever to play, a poor internet connection will not hinder the players' enjoyment of the game.

The experiment that was chosen to test the Unity game was a survey. Several participants were selected to play the game for at least five minutes. Then, they were asked to fill out a Google Forms survey that asked two questions. The first question was "How do you rate the player movement of the game?", and the second question was "How do you rate the overall enjoyment of the game?". Each of the questions would provide the participant with a scale from one to ten to choose their rating. After answering these two questions, the participants would also have the opportunity to fill out an optional free-response section that asked the users if they had any other feedback to provide. This would offer participants the ability to express their thoughts in a way that would not be possible by only using the previous two questions.

With this experiment, a high overall score regarding the player movement indicates that

controlling the character was a smooth and intuitive experience. A high overall score regarding the general enjoyment of the game could also indicate that members of the general public would be willing to play more games like this in the future. By taking these two scores, we can also identify any possible issues with the game. For example, if the player movement was rated highly and the overall enjoyment was rated much lower, this could imply that other areas in the game are lacking. On the other hand, if the player movement was rated much lower than the overall enjoyment, the general public might prefer a different player movement implementation.

The remainder of the paper is split into five sections, which will be labeled 2 through 6. Section 2 details the difficulties faced when trying to design and implement the game as well as come up with an experiment to test the game. Section 3 explains how the general implementation of the Unity game was done, and more details are provided on particular aspects of the game such as the player movement. Section 4 brings up how effective the Unity game is at providing its players with an enjoyable gaming experience by surveying participants on the intuitiveness of the player movement and the overall enjoyment of the game, and Section 5 introduces a few related works and states how they compare to this work. Section 6 provides a conclusion that includes a summary of the application, some current limitations to the application, and what can be done in the future to resolve these limitations.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Thinking ideas of the setting and the Design for the Game

There were many challenges while building the game, and one major challenge was thinking of ideas for the game, specifically with the setting of the game. There are lots of elements in the game such as the movement, the map, and the game's overall experience that should be taken into consideration. First, the character in the game was created using 3D objects in Unity. Then, after brainstorming how the map would look and what the scene should be based on, a city was selected to be the environment in which the player would be placed. To build the city environment, city buildings were selected from the Unity Assets Store to be imported into the project. Creating the map in the game was a lengthy process, as the importedbuilding objects had to be placed in a specific manner to resemble a modern city in the real world. Lastly, the buildings in the city were colored with different textures. Besides the settings, there are many differentsmaller aspects in the game that had to be considered in the design of the project.

2.2. Implementing Health into the FPS game

The second challenge that was faced when making the FPS game was implementing health into the shooter game. First, both the player and the enemies needed to have health so that they would die after they are attacked enough times. A script was written that determined how much health a player and each of the enemies would have. However, with the implementation of the game, the enemies would not have any weapons and would not be able to deal damage with far-range attacks. Therefore, the only way that the enemies could deal damage was by walking into them. The enemies would have a certain amount of health and would die when the character used the gun to shoot the enemy in the head once. The user's screen would have a dotted crosshair and when left clicked, the gun would shoot and subtract the health from the enemy. An end screen was later designed in which if the character dies, a screen would appear showing the words "Game Over" and a quit button to close the game.

2.3. Implementing Shooting in the FPS Game

The last challenge that was faced was implementing shooting in the shooter game. A model of the gun first had to be chosen from the Unity Assets store, then the gun was imported into the game. Then, a crosshair was created in the character's first-person point of view, and a good angle was carefully chosen to place the gun in the character's hand. A script was written to control what will happen when the gun shots. The gun was designed so that when left clicked, the gun would fire and do damage to enemies if the crosshairs are placed directly on the enemy. The gun will take away one health from the enemy if it hits the enemy, while the character will also lose health if the enemy touches him. The shooting was implemented into the game in the code by returning a Boolean value after a target has been successfully hit.

3. SOLUTION

The game is created using the Unity engine, and the game has three main screens. The first one is the main menu screen, which provides the player with two buttons; one button says "Play" and brings the player into the game screen, while the other button says "Quit" and closes the game when pressed. Once brought into the game, the player will spawn into a city scene, and the user is equipped with a gun. The player can find enemies that spawn around the map and pursue the player. The player can shoot the enemies in the head to defeat them and make them disappear from the map. However, if the player lets the enemies come into contact with their character for too long, the player will lose and the game over screen will appear. From here, the player can choose to either play again using the respawn button or quit the game using the quit button.

Objects were a major part of the Unity game and are responsible for much of the game implementation. The city buildings, the enemies and the player's character, and even the buttons on the main menu and game over screen would not exist without using Game Objects in Unity. To provide the functionality to the game, C# scripts were written and attached to certain objects in the game. [12]These scripts would be able to control various aspects of the game, from the movement of the character to the implementation of shooting and player/enemy health.

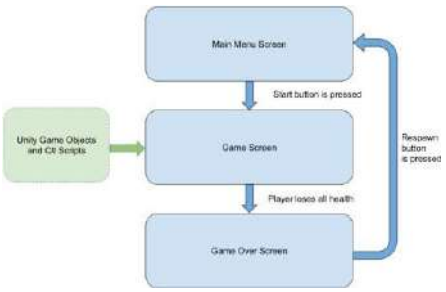


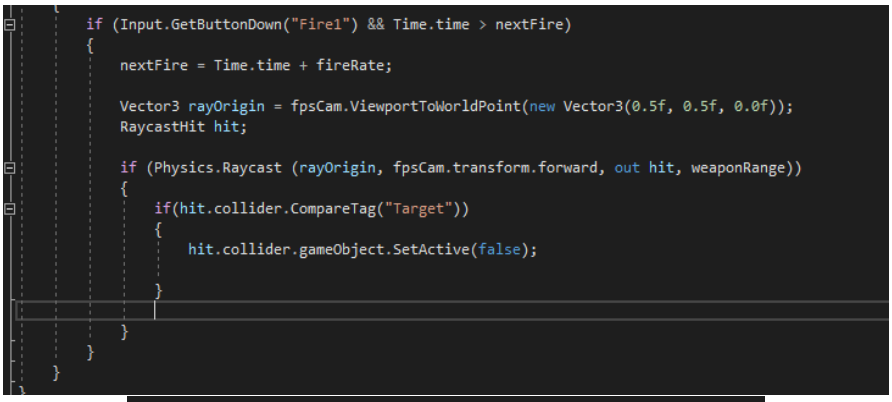
Figure 1. Overview of the solution

The shooting in the game was implemented by using Unity Raycast, which is the Raycast system where data is returned after a target is successfully hit by a ray. With the shooting implementation, a point was added to the screen as a crosshair, in which the crosshair is a ray in the system. In a C# script, Unity would first check whether the button to fire the weapon (left mouse click) was being pressed and whether enough time has elapsed between now and the last time that the player fired the weapon using if statements. The way that this was implemented was by creating variables called nextFire and fireRate. The fireRate was a set number, while nextFire would add the current time and the fireRate to determine when the next time that the player could fire was. By doing so, a hard limit could be set on how often the player could fire the weapon. An

implementation like this could prevent people from having an advantage from clicking faster and incentivize them to click as rapidly as possible. Without such a limit, some players may use auto-clickers to cheat; an auto-clicker is a form of computer software that can automate the process of clicking a mouse, and many of them have settings for the users to freely adjust how quickly the clicks occur. Alternatively, other people who play legitimately may suffer carpal tunnel syndrome or other injuries or strains related to the fingers and hands by clicking too much and too quickly. Therefore, this fixed firing rate system can allow people to play on equal grounds. After the object has been hit, the data would be stored in a Raycast variable and could later be used to determine how much health they have remaining.

The Raycast shoot system, which is done in the Update method, involves first getting the data back to see if an enemy had been hit by checking whether the collision involved an element with the tag of “Target”. [11] If an enemy was shot in the head, the enemy would disappear by setting the game object to inactive, as the enemies in the game only have enough health to be defeated in one shot. On the other hand, the player has 100 health and loses 10 health with every collision with an enemy, enough to endure 10 collisions. If the character reaches 0 health, the game will end and the game over screen would be displayed. The value returned can later be used in the health system to determine the health of the enemies. Enemies were coded to spawn randomly around the map so the player wouldn't face lots of enemies in one direction at once.

The C# script responsible for enemy spawning was implemented to have one enemy spawn every ten seconds. A pause menu was later created in the game where the user can click “ESC” to pause the game. The get key function was used to check if the user had clicked the pause button, then the time scale function would play a role in pausing the game, because the time scale would change to 0 if the game is paused and to 1 when the user exits out of the pause menu. The pause menu has a quit button, and if the user clicks the button, the game would be exited. Player movement was added to the game by using a MonoBehaviour class. First, a collision script was made where players wouldn't fall through the ground when they are walking on it. Then the sensitivity was set for the player's walk speed, player run speed, player jump height, and crouch height. The movement keys are set similarly to other games, “WASD” to move around and space to jump. In many current FPS games, the movement speed of the player is brought to only a fraction of the original movement speed when crouching (which can be done by multiplying the player’s speed by a certain amount), as this mimics how quickly a person would move when crouching in real life and allows for more precise movement. However, as this game does not require precision in character movement, the implementation of crouching involved the movement speed staying consistent and uninterrupted, which may create smoother player movement.

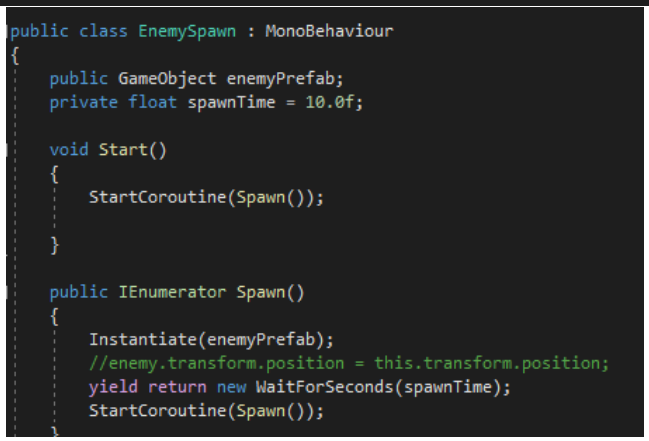


```
    if (Input.GetButtonDown("Fire1") && Time.time > nextFire)
    {
        nextFire = Time.time + fireRate;

        Vector3 rayOrigin = fpsCam.ViewportToWorldPoint(new Vector3(0.5f, 0.5f, 0.0f));
        RaycastHit hit;

        if (Physics.Raycast (rayOrigin, fpsCam.transform.forward, out hit, weaponRange))
        {
            if(hit.collider.CompareTag("Target"))
            {
                hit.collider.gameObject.SetActive(false);

            }
        }
    }
}
```



```
public class EnemySpawn : MonoBehaviour
{
    public GameObject enemyPrefab;
    private float spawnTime = 10.0f;

    void Start()
    {
        StartCoroutine(Spawn());

    }

    public IEnumerator Spawn()
    {
        Instantiate(enemyPrefab);
        //enemy.transform.position = this.transform.position;
        yield return new WaitForSeconds(spawnTime);
        StartCoroutine(Spawn());
    }
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

[RequireComponent(typeof(Collider), typeof(Rigidbody))]
public class PlayerMovement : MonoBehaviour
{
    //Look
    public float sensitivity = 1f;
    public float smoothing = 2f;

    private Transform charCamera;
    private Vector2 mouseLook;
    private Vector2 mouseDelta;

    //Move
    public float walkSpeed = 5f;
    public float runSpeed = 10f;
    public KeyCode runKey = KeyCode.LeftShift;

    private Rigidbody rb;

    //Jump
    public float jump = 5f;
    public KeyCode jumpKey = KeyCode.Space;

    private bool isGrounded = true;

    //Crouch
    public float crouch = 0.25f;
    public KeyCode crouchKey = KeyCode.LeftControl;

    private float normalYposition = 1f;

    void Start()
    {
        //Look
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible = false;
        charCamera = Camera.main.transform;

        //Move
        rb = GetComponent<Rigidbody>();

        //Crouch
        normalYposition = rb.transform.localScale.y;
    }
```

Figure 2. Screenshots of the FPS game's code

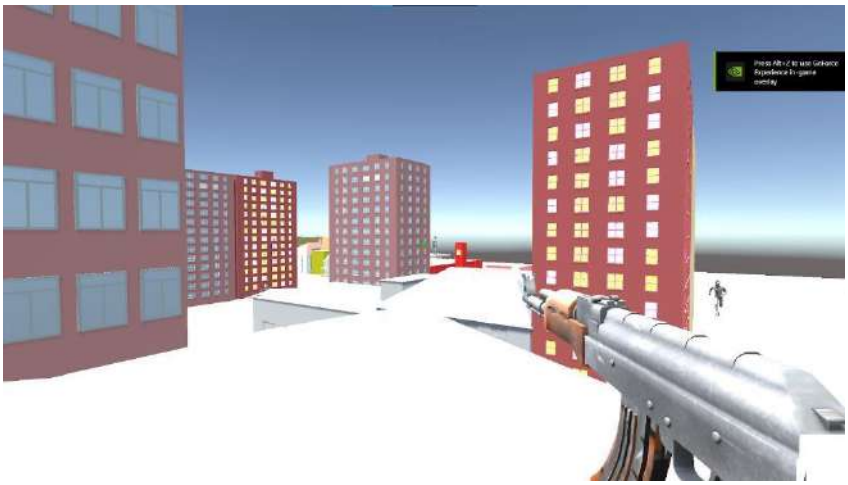


Figure 3. Screenshots of the FPS game

4. Experiment

4.1. Experiment 1

The experiment that was selected to test the effectiveness of the implementation of player movement in the game as well as the effectiveness of the game as a whole at providing players with enjoyment and entertainment is a Google Forms survey [14]. First, fifteen participants were individually given a ZIP folder containing the first-person shooter game. After unzipping the folder and opening the game, they would play the game for at least five minutes. Before playing the game, each participant was given the message “Try to test the movement of the game as much as you would like.” Because the participants were given the same version of the game and the same message, this reduces any confounding variables in the experiment. The first question that the survey asks is “How do you rate the player movement of the game?” and a scale from one to ten is provided for the participants to answer.

Participant Number	Rating of Player Movement
1	9
2	10
3	7
4	6
5	8
6	8
7	9
8	7
9	9
10	10
11	10
12	7
13	9
14	8
15	7

Figure 4. Table of participants

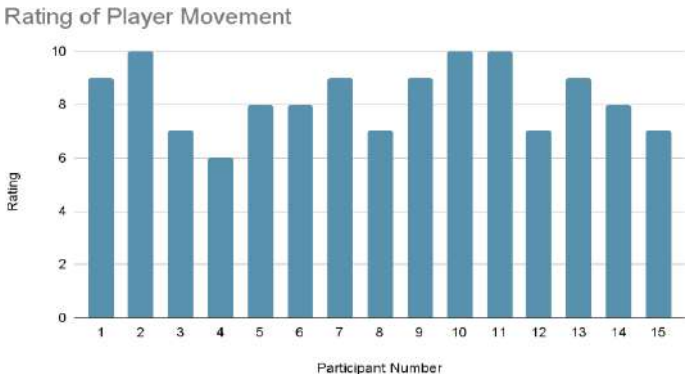


Figure 5. Rating of Player Movement

According to the results, it appears that the participants rated the player movement in the FPS game relatively high overall. The lowest rating of 6 was only given by one participant, while the highest rating of 10 was given by three participants. These ratings could indicate that participants found the implementation of the game's player movement to be done well. This idea is backed up further by the optional feedback that some of the participants provided. Participants stated that the player movement seemed smooth, intuitive, and uninterrupted when they played the game. In particular, a few participants pointed out that crouching while moving would not reduce the momentum, and this was a feature that they gladly welcomed. However, one of the participants felt the opposite and believed that since the player movement was implemented so much differently from other traditional modern FPS games, not slowing down when crouching was jarring.

4.2. Experiment 2

The second question in the survey is "How do you rate the overall enjoyment of the game?". To keep the survey consistent, a scale from 1 to 10 was also provided for the user to answer. At the end of the Google Forms survey, an optional free-response section was provided to the participants. In case they have any feedback that would not be possible to provide from the two previous questions or would like to expand upon something from the previous questions, they would be able to do so. Since there are fifteen participants, sample size is large enough to account for variability.

Participant Number	Rating of Player Movement
1	9
2	7
3	8
4	5
5	6
6	7
7	8
8	7
9	9
10	10
11	8
12	6
13	7
14	8
15	7

Figure 6. Table of participants

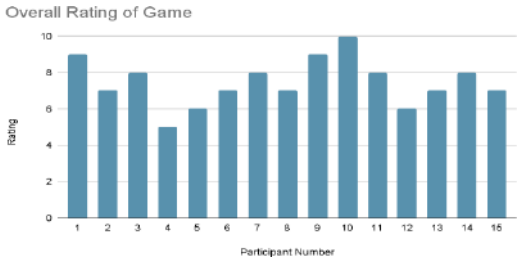


Figure 7. Overall Rating of Game

The responses to the second question were relatively positive. One participant gave the highest score of 10, while another participant provided the lowest score of 5. The results indicate that the participants were moderately pleased with the entertainment value of the game overall. A better explanation of the results can be seen in the feedback provided in the free-response section. While most participants stated that playing the game was fun, there were several issues that still had to be addressed. The largest issue among them was the shooting mechanic. While shooting in the game worked as normal and players could defeat enemies by shooting them enough times, there was no visual or audio cue to indicate that the player was shooting. Some participants stated that they thought the shooting mechanic did not work at all as a result. While the player movement in the game may be polished, the other features of the game did not match that same level of polish. In the first question that measured how well the player movement was implemented in the game, participants generally seemed to agree that the movement of the player was fun, intuitive, and somewhat unique to traditional FPS games. The second question that gauged the enjoyment of a game as a whole did not receive responses as positive as the previous one, since the participants found issues in quality in aspects of the game besides player movement that reduced the entertainment value of the game.

Taking both results into account, the game appears to be much more well-received with its player movement than as a whole. This was to be expected, as the player movement was one of the primary aspects of the game that was emphasized during its development. However, this could also indicate that the other areas in the game are lacking, such as the shooting mechanic or the enemy spawning mechanic.

5. RELATED WORK

A first-person shooter game was developed to study the effects of implicit and explicit biofeedback, where explicit biofeedback is consciously displayed and/or felt while implicit biofeedback is felt at the subconscious level. The study concluded that explicit biofeedback had a much more significant impact on the players than implicit biofeedback did and indicated that explicit biofeedback interaction may have a bright future in video games [1]. Both the study on biofeedback and this work focus on first-person shooter games as a primary topic. While the study on biofeedback had a stronger focus on the biosensors' data for results, this research places an emphasis mainly on the players' perceived experience regarding character movement in a newly developed FPS game.

Another study evaluates how high players of a first-person shooter game believe the quality of the gameis and creates a model using ping, jitter, and packet loss values of players to predict the perceived quality of the game from other players. The model from this research indicates that the perceived quality of a game has a high correlation with reduced amounts of ping and jitter values [2]. While the study involving a perceived quality model focuses on how poor connection issues can impact the players' experience of a game, this work emphasizes adjustments in the implementation of player movement to enhance players' experience in FPS games.

A related work presents a study regarding how player performance and enjoyment in first-person shooter games are affected by frame rates. According to the results of the study, the number of frames per second that a player experiences greatly affects their performance in all aspects of the game (including both movement and shooting), but the improvement of the player becomes smaller the higher the number of frames per second becomes [3]. This related work is similar to our work in that the experience of players in FPS games is tested, but the related work emphasizes frame rate to do so while this work utilizes development choices regarding the player movement to do so.

6. CONCLUSIONS

Our method of finding a way to improve player movement in first-person shooter video games is creating a game that implemented player movement in a different manner. The most notable difference from traditional FPS games is that when crouching, the momentum of the player remains unchanged. In the game, the player is free to jump on top of buildings in a city and shoot large enemies to defeat them. The game is currently endless, meaning that the player's goal is to survive as long as possible and stop the enemies from walking into the player and dealing contact damage [13].

To test the effectiveness of the player movement in the game at maintaining player interest and providing an enjoyable gaming experience, an experiment was conducted in which fifteen participants were gathered to play the game and test the controls, then fill out a Google Forms survey that asked the participants how much they enjoyed the player movement and the game as a whole. From the results, the player movement was rated higher than the enjoyment of the overall game. While this indicates that the player movement implementation was done well, the other areas of the game do not seem to match that level of quality. The feedback also reinforced this notion, as the shooting mechanic appeared to be a feature that reduced the level of enjoyment from most participants who provided feedback. Nevertheless, the maintaining of momentum while crouching was widely well-received, which can indicate that such an implementation in future games may prove successful in the right circumstances, such as games that do not require precision in movement.

One of the biggest limitations is that the game currently does not provide an indicator of when the gun has been fired. The game can still register if the player shoots and hits enemies, and the enemies disappear once they have taken enough shots. However, a new player may not know this, and they may assume the game is bugged instead. If there is no visual or audio cue that indicates when the gun has been fired, the player may not have the information they need to make the correct decisions in the game. The lack of effects may also ruin the immersion of the game for some players.

An effective way to solve the issue is adding a sound effect whenever the gun has been fired. This will let the player know that the gun has been fired. Eventually, a visual effectlike a flash can be used as well [15].With more effects and polish to the game, the players can become more immersed.

REFERENCES

[1] Kuikkaniemi, Kai, et al. "The influence of implicit and explicit biofeedback in first-person shooter games."Proceedings of the SIGCHI conference on human factors in computing systems. 2010.
[2] Wattimena, A. F., et al. "Predicting the perceived quality of a first person shooter: the Quake IV G-model."Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games. 2006.
[3] Claypool, Kajal T., and Mark Claypool. "On frame rate and player performance in first person shootergames." Multimedia systems 13.1 (2007): 3-17.
[4] Voorhees, Gerald A., Joshua Call, and Katie Whitlock, eds. Guns, grenades, and grunts: First-person shootergames. Bloomsbury Publishing USA, 2012.
[5] Hew, KheFoon, and Wing Sum Cheung. "Use of three-dimensional (3-D) immersive virtual worlds in K-12 and higher education settings: A review of the research." British journal of educational technology 41.1 (2010): 33-55.
[6] Shafer, Daniel M. "Causes of state hostility and enjoyment in player versus player and player versus environment video games." Journal of Communication 62.4 (2012): 719-737.
[7] Pirker, Johanna, et al. "Analyzing player networks in Destiny." Entertainment Computing 25 (2018): 71-83.
[8] Jennett, Charlene, et al. "Measuring and defining the experience of immersion in games." International journal of human-computer studies 66.9 (2008): 641-661.
[9] Strömberg, Hanna, Antti Väätänen, and Veli-PekkaRäty. "A group game played in interactive virtual space: design and evaluation." Proceedings of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques. 2002.
[10] Breuer, Johannes, et al. Violent video games and physical aggression: Evidence for a selection effect amongadolescents. Vol. 4. No. 4. Educational Publishing Foundation, 2015.
[11] Niu, Hanlin, et al. "Accelerated sim-to-real deep reinforcement learning: Learning collision avoidance fromhuman player." 2021 IEEE/SICE International Symposium on System Integration (SII). IEEE, 2021.
[12] Konzack, Lars. "Computer game criticism: A method for computer game analysis." CGDC Conf.. 2002.
[13] Adiwikarta, Rendy, and HaryaBimaDirgantara. "Pengembanganpermainan video endless running berbasis android menggunakan framework game development life cycle." Indonesia: KALBIScientia, ISSN (2017): 2356-4393.
[14] Argyris, Chris, and Donald A. Schon. Theory in practice: Increasing professional effectiveness. Jossey-bass, 1974.
[15] Lv, Zhihan, et al. "Game on, science-how video game technology may help biologists tackle visualization challenges." PloS one 8.3 (2013): e57990.