Exploring the Impact of Real-World Consequence on a Player's Experience in Virtual Reality

Jack Cooper
University of Technology Sydney
jack.cooper7776@gmail.com

Abstract

This paper explores the impact that high consequence feedback has on a player in VR, and the behavioural changes of the player while playing a game with and without the feedback. Virtual Reality being a highly immersive system, it is an appropriate format to test this kind of feedback and its effects. The paper outlines the process taken to develop a system to provide high consequence feedback, and interfacing that with a VR game developed to test the system. The work undertaken in this paper hopes to outline the effectiveness and implications consequences have on the experience of video games.

KEYWORDS

Gaming, Consequences, Player, Virtual Reality, VR, Electrocution, Electric, Shock

1 Introduction

Recently Virtual Reality has come into the spotlight, boasting to have far higher levels of immersion than traditional gaming, however, this highlights the question: what is immersion and what factors contribute to the "higher" or "lower" levels of it. Immersion is often a core aspect of video games, often requiring careful design of majority of a games system, including UI, gameplay and importantly, feedback. Feedback is usually delivered to a player via the screen, vibrations and audio; but can other types of feedback increase or take away from a player's experience or immersion. Feedback to a player can provide enhancements to a game's theatrical components e.g. audio to enhance a story or dialogue, and enhance a game's gameplay components e.g. vibrating the controller when damage is taken.

Virtual Reality Electric Shock Feedback

Currently haptic feedback to a player in a system designed for immersion (VR) is relatively absent aside from vibrations in the controllers. How will giving high consequence feedback to a player via a small electric shock enhance or subtract from a player's enjoyment, immersion and in-game strategies. Similar to the difference between laser tag, being relatively painless and paintball, which requires strategy to avoid pain, this will explore how behaviour of a player changes to high and low consequences.

2 Related Work

The use of pain as a gameplay mechanic is not new, a number of examples can be found in *Games of Pain: Pain as Haptic Stimulation in Computer-Game–Based Media Art* [1], where they discuss a number of different systems created for new and existing games. One example is *"Tekken Torture"* where the players face off against each other in the fighting game Tekken-Tournament and each time their character takes damage, they are given a shock via electrodes placed on their right arm. This caused the players to create a deeper connection with their character as they also feel the consequences of losing.

High consequences in gaming has also become a source of entertainment as see on YoutTube, where is it becoming increasingly common to see people building systems like this to increase viewership. Although these videos show the effect that high consequence to a player has on their experience while playing a game. In *I Get Shocked When I Lose Hearts in Minecraft...* [2] the player with the shock device prioritises not taking damage far higher than in usual gameplay, but this also creates new strategies and teamwork e.g. avoiding combat at all costs, instead having his friend be his bodyguard. The system used in this video is similar to the feedback device created for this project.

3 Approach

In order to determine the impact that high consequence feedback on a player's experience 2 core systems need to be created. (1) A system to provide the feedback without intruding on the other common feedback systems. (2) An application that utilizes the feedback system in a meaningful way.

The feedback system would need to meet a number of requirements to be useable:

1. The system is safe to use. The system should be able to provide high consequence feedback to a player without harming them.
2. The system should be portable and light weight. The system will be used while in a VR experience, so in order to keep with the freedom of movement that VR requires, the system should be wearable without hindering movement or being obtrusive.
3. The system needs to be comfortable. This is required as to not take away from the

immersion, the player needs to be able to forget that the system is there.

4. Maintain consistency. The system needs to be able to provide the feedback whenever required, as without consistency the player may struggle to make connections to what is causing the feedback and ultimately negate the intentions of the system.

The application also needs to meet a number of requirements to properly utilize the feedback system:

1. Works in VR. The game is intended to be an immersive experience to test how the feedback affects that.
2. Have multiple gameplay opportunities. Part of the question is to see how a player's actions and strategies change with the feedback; so, in order to compare, the player needs to be able to adapt and play differently if needed.
3. The game needs to utilize the feedback system in a meaningful way, otherwise the feedback will have no impact on the player or game other than confusion.
4. The game needs to be able to be played with and without the feedback to be able to compare how the experience changed.

Once all requirements are in place, participants will play the game with and without the feedback system and will give insight into how it made them play and feel about the experience.

4 The System and Game

The system to provide the feedback is made up of 3 components, the device to shock the player, the device that controls the shock and the system to trigger the controller device.

4.1 Shocking Device

The shocking device is made of a dog training collar. This was modified to be more comfortable than how it came as the contacts tend to "jab" the wearers arm quite a lot. The comfort modification took the form of running leads from the contacts to two contacts on a Velcro wrist band.

Figure 1: Annotated image of the electric shock device modified to be more comfortable.

This had to be done twice as there is a shock device for each side of the body.

4.2 Remote Control

The remote for the shock device was the next system to be created. The base remote was manually operated only with buttons for changing: mode (Vibrate, Shock, Beep), change level of shock/vibrate, change channel (allows swapping between 2 separate shock devices (collars)), and the trigger button which triggers the selected mode on the shock device. This needed to be modified to allow Unity to control the device. An Arduino Uno was used as the interface between the remote and a PC. The use of perf board, transistors, diodes, jumper leads and header pins allowed me to create this shield that sits atop the Arduino.

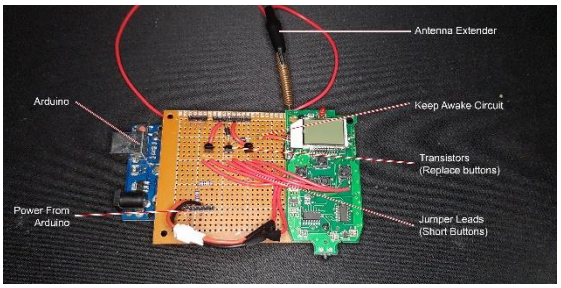

Figure 2: Annotated image of the modified remote atop an Arduino Uno.

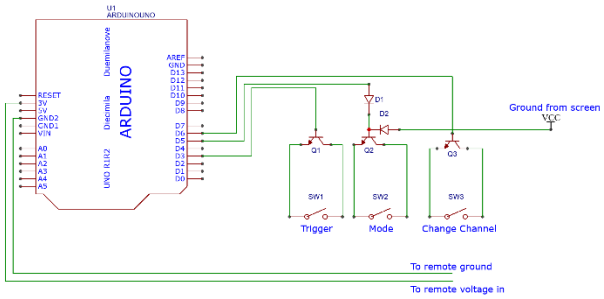

Figure 3: Wiring schematic for the Arduino shield to the remote board.

The transistors stand in for the buttons on the remote, as when the base pin is powered, it allows current to flow between the collector and emitter. This is essentially the same function of a button, except controlled with current. Leads from the collector and emitter are connected to each side of the buttons, creating a parallel path for current to flow. It is worth noting that the buttons still function with this setup, allowing manual control if necessary.

Two issues were encountered when converting the controller to this state. The first being the range of the remote seemed to have dropped significantly, to a point where it was unusable more than a few centimetres away. The solution to this was accidentally found while trying to debug the issue. When checking for voltage on the antenna with a multimeter, the device began to work. I discovered that the unshielded cable on the multimeter probe was acting as a larger antenna, so a crude antenna extender was made to solve the issue.

The second issue was that the remote had an internal timer that would put the remote to sleep after 1 minute of inactivity and turn back on when the mode select button was pressed. My initial plan was to have unity send a signal to "press" the mode button after 1 minute if it hadn't sent a command in that time. However, this created the possibility of desync between what Unity believed the settings were and what the settings actually were if not timed correctly. The workaround that was implemented was that the display ground cable would output 0 volts when on, and when the screen turned off the voltage from ground would jump to around 0.4v. This meant I could use that voltage to trigger the already existing transistor for the mode select. This solution works perfectly; whenever the remote goes to sleep, the screen ground cable activates the transistor and triggers the mode select which turns it back on. All this take a fraction of a second and causes no desync between Unity and the remote. All of this

addresses the issue of ensuring the system is consistent, one of the requirements outlined above.

4.3 The Game and Output to the Remote

The interface between Unity and the Arduino was achieved using an asset package from the Unity store. The package was Kreation.Arduino - On/Off Controller [4] which allowed me to control the Arduino using the Unity Animation Timeline and Animator. Animations were created where a keyframe would set a pin on the Arduino to high/low, and a script managing player damage would ask the Arduino manager to play the clips when necessary. The animation clips had to be very fast in order to provide very short shocks, so each animation was only a few frames long. Unfortunately, the asset package didn't provide the functionality to control the Arduino in script other than asking the animator to play the animations. The asset package worked by utilizing Arduino's standard Firmata [3] which is a protocol to allow serialized communication between a PC and Arduino.

The game was created for virtual reality also using asset packages: VR Shooter Kit [5] and SteamVR Plugin [6].

SteamVR Plugin was used to make the game launchable using SteamVR to run the game. The VR headset used was a Lenovo Windows Mixed Reality headset, which was able to use SteamVR to play the game.

VR Shooter Kit was used to create the gameplay and environment, providing easy setup, controls and tools for VR. The easy setup of player locomotion was a huge help, and being able to get the VR controller input from these systems was far easier than trying to navigate the SteamVR code as it is not commented very well. The other tools including weapons and grabbing, sped up development a large amount as the game was going to be based around shooting. This left more time to develop the enemies which required a lot more balancing than usual due to the high consequence feedback.

The game created was built with the idea that the player would not want to be electrocuted constantly. This meant the enemies had to be balanced in a way that meant the player could out manoeuvre them if needed. The enemies' movement, aiming and shooting is largely physics based. Turning towards a player uses AddTorque() towards the players direction, to create a floaty turning circle and movement used AddForce() so that they would overshoot their target location. The enemy aiming was also done with physics; the enemy eyes worked on a lookat system, with an empty game object suspended from another with a spring joint. The look at target would have a force applied to it when the player is in range, meaning the look at target would move in the players general direction, but not perfectly, additionally enemy movement would cause the look at target to swing around even more. The enemy would simply fire their weapon towards the lookat target instead of the player. This caused the enemy's to be inaccurate enough to not hit every shot, but accurate enough to land enough shots that the game would work. The enemy projectiles were also physics objects that had a bullet drop curve and meant that they couldn't hit you from very far away, also adding to the inaccuracy of the enemies.

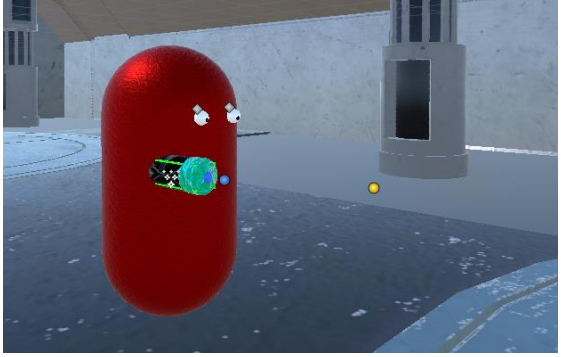

Figure 4: An enemy looking at it's look-at target (in orange)

The last important system created was the player hitboxes. There was one for each side of the body, attached to the "player pockets" object. This object was simply an object that would follow the players position, but not rotate on any axis other than the y axis. This meant it would face the same way as the player but would not rotate in unwanted ways. These hitboxes were the ones that asked the Arduino controller to trigger functions on the remote. There was one hitbox for each side of the players body so that the player would get shocked based on where they were being shot from, adding more function to the feedback other than consequence. This application of the feedback meant that it was being used in a meaningful way, by providing information to the player, it has purpose, fulfilling one of the requirements of the game outlined above.

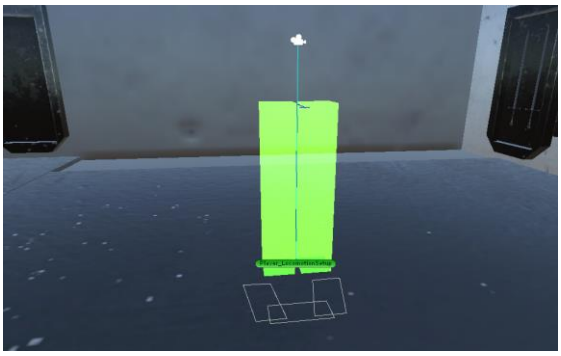

Figure 5: The player hitboxes highlighted in green, one for each side of the players body.

There were some safety systems implemented that allowed the user to specify a minimum time between electric shocks in order to limit the amount of feedback if the player didn't feel comfortable being constantly shocked. In addition to this safety buttons were added so that if a player had, had too much, they could hold down one of these buttons and completely prevent the requests for shocks from triggering.

5 Tests

Tests consisted of participants playing the game with and without the shock system for comparison. Most participants played with the shock system on first then switched to without in order for them to be able to compare how the system affected their time in the game.

In total there were 4 participants and each completed the test then completed a survey that was built to find out how the feedback changed their behaviour, enjoyment and fear.

6 Results

Participants were asked on a scale from 1-10 how willing would you be to play again (using the shock system).

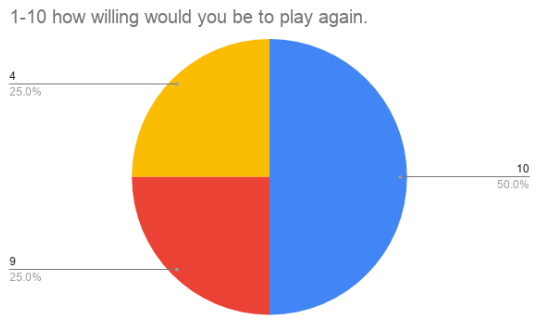

Figure 6: Pie chart breaking down the responses to "1-10 how willing would you be to play again"

50% of players answered 10/10, and one even returned later to play again. Another 25% said 9/10 leaning towards similar results. 25% showed hesitation to try it again, although the participant that replied with a low chance to play again also noted that they were hesitant to try it in the first place.

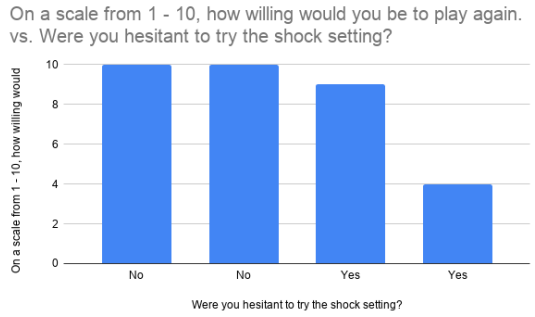

Figure 7: Column chart breaking down the responses to "Were you hesitant to try the shock setting" vs "1-10 how willing would you be to play again"

This chart demonstrates that at least one person that was hesitant to try the game in the first place still gave a high chance to play again.

Responses to the question "Did you notice the feedback changing how you play? If so, how?" gave a lot of insight into how the feedback changed the players experience. One participant wrote *"It made me much more cautious in where I had to be. The fear of pain gave it extra incentive."*. This is fairly in-line with the other responses given for that question, where the consequence caused the player to actively think about their actions and strategies in order to avoid getting shocked.

Figure 9: Participant getting electrocuted

After a test many of the participants spoke about the differences between having the shock system on vs off. The common theme was that the game was boring, had no objective and no real threat while the system was off, however, with the system on, it felt as if there was a goal and threat. This points towards the idea that without conflict, the entertainment value of a game is far lower and that the shock system is able to provide a layer of conflict to a game even if the core gameplay is considered "boring".

During the tests there were people watching the participant take the tests. A lot of the time many members of the audience found the tests fun to watch, with a lot of humour and talk of applications for the system, however, when the tests were without the shock system, the room was far quieter and the test less entertaining. This reinforces that the shock system can provide entertainment, but not only to the player, but to those watching it. Similar to the increase in YouTube videos with similar systems being created e.g. *Michael Reeves, A Robot Shoots Me When I Get Shot in Fortnite* [7] and *Dream, I Get Shocked When I Lose Hearts in Minecraft...* [2].

Many of the participants and audience spoke frequently about the application of the shock system to multiplayer games and how that could enhance the experience. This is explored in *Pau Waelder Laso, Games of Pain: Pain as Haptic Stimulation in Computer-Game–Based Media Art* [1] specifically in "Tekken Torture", as players are inflicting pain on each other, connecting a player with their avatar, limiting their abilities while being in pain and pushing players to be more competitive.

One last interesting statistic was that all players that gave a high probability of wanting to paly again all claimed they didn't use or intend on using the safety "stop shock" buttons, but highly valued their inclusion.

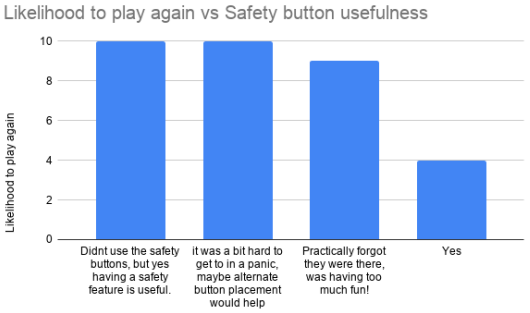

Figure 9: Likelihood to play again vs Usefulness of the safety buttons

7 Discussion

7.1 Lessons learnt

The results show that there is a place for high consequence feedback in gaming, mainly for gameplay applications to keep control in the players hands. High consequence feedback can provide a layer of entertainment to games for players and audience even if the actual gameplay is limited as the fear of consequence can be a large enough driving force on its own for a player.

High consequence gameplay is not for all people, but for those that are willing to try it and enjoy it, it can provide far greater depth to games if used correctly.

7.2 Limitations

Due to time constraints and it being somewhat difficult to find participants for the tests, not much data was collected. I believe that the data is just an indication that it is possible that this kind of feedback does provide something to a game, but there is too little data to determine to what extent and how often.

It is also to be noted that the game was lacking in gameplay, therefore, although the electrocution added to the experience, the experience was relatively empty as it was (although this could have highlighted the

effectiveness of bringing entertainment to a game with high consequences).

Originally, I wanted to measure each participants heart rate during the test. This was unviable due to the cost of sourcing a heart rate monitor, and implementing it into the tests would have taken up a significant amount of the testing time.

7.3 Future Work

In future exploration of this topic, I believe that more tests should be done with more participants. Additionally, more games and types of games with varying difficulty/gameplay experiences in order to better gauge how the consequences affect the player and their experience in different situations.

It would also be interesting to get the members of the audience's input on the subject as they may be able to provide insight into the entertainment factor for a nonplayer and the potential applications in that field.

8 Conclusion

Building the systems was a huge success, the shock system worked consistently and provided the high consequence feedback that was intended. Players found this feedback to be both useful and as a driving force for the gameplay and many saw it as having a place in the gaming industry. However, with the limited sample size it is hard to takeaway solid lessons from the project.

References

[1] Pau Waelder Laso, 2007, Games of Pain: Pain as Haptic Stimulation in Computer-Game–Based Media Art, <https://search.lib.uts.edu.au/permalink/61UTS_INST/1vd9sba/proquest1035869691>

[2] Dream, 2020, I Get Shocked When I Lose Hearts in Minecraft..., Viewed 19/02/2020, <https://youtu.be/AA7AE-3yq7Y>

[7] Michael Reeves, 2018, A Robot Shoots Me When I Get Shot in Fortnite, Viewed 30/08/2018 <https://www.youtube.com/watch?v=D75ZuaSR8nQ>

Unity 2019, Unity, Game Engine, 2019.2.12f1, <https://unity.com/>

[3] Arduino 2019, Firmata, Open Source <https://www.arduino.cc/en/reference/firmata>

[4] Lokel Digital, Kreation.Arduino - On/Off Controller, 0.1.0.1, <https://assetstore.unity.com/packages/tools/input-management/kreation-arduino-on-off-controller-151730>

[5] Unity3DNinja, VR Shooter Kit, 1.36, <https://assetstore.unity.com/packages/templates/systems/vr-shooter-kit-152587>

[6] Valve Corporation, SteamVR Plugin, 2.5.0, <https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647>