Replay to Play: Creating Accessibility in VR with Reused Motion Input

A Technical Report
presented to the faculty of the
School of Engineering and Applied Science
University of Virginia

by

Cody Robertson

May 8, 2020

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

Signed: _____

Approved: _____ Date _____
Seongkook Heo, Department of Computer Science

Replay to Play: Creating Accessibility in VR with Reused Motion Input

Abstract

Existing virtual reality (VR) games and applications tend not to factor in accommodations for varied levels of user ability. To enable users to better engage with this technology, a tool was developed to record and replay captured user motion to reduce the strain of complicated gross motor motions to a simple button press. This tool allows VR users with any level of motor impairment to create custom recordings of the motions they need to play VR games that have not designed for such accessibility. Examples of similar projects as well as recommendations for improvements are given to help round out the design space of accessible VR design.

Introduction

In many instances, high-end in-home virtual reality is synonymous with a head-mounted display (HMD) on the user's face and motion-tracked controllers, simulating the hand's ability to grip and hold objects, in a user's hands. This is the case with all forms of consumer available HMD that is driven by a traditional computer rather than an integrated computer, including the Oculus Rift, Valve Index, HTC Vive, the variously produced Windows Mixed Reality HMDs, and Playstation VR with the Move Controllers. While the two technologies can be combined to produce a high-fidelity immersive environment, the reliance of gross motor motion-tracked controllers to interact with the virtual world restricts this technology to those who are either able-bodied or those willing to interact with limited applications. VR game developers are currently maturing their techniques for building more complicated but comfortable experiences. The phrase "comfortable VR experiences" currently mean that a user will not experience motion sickness during a reasonable period of use, not that the essential motions required for the experience will not be strenuous or will be able to be modified for ability level, similar to how push-up can be modified to be done on the knees rather than the ball of the foot. While developers are designing for one type of accessibility by trying to prevent motion sickness, they de-emphasis any other sense of the concept of accessibility. This project aims to overcome this issue by empowering users to create and store custom motions in a safe environment to use later in the games of their choice.

Background

User interface developers have attempted to address accessibility in a variety of ways. One way is through modification - making additional improvements, features, or accessories to an existing product. This approach situates accessibility as an accommodation or afterthought, creating new specialty features or additional products to meet the needs of excluded audiences. It can be an efficient method, as it allows developers to add onto existing products and focus on expanding their existing audiences instead of competing with them (Wobbrock et al., 2011). However, these

modifications could also have limitations in their ability to meet user's access needs. Similarly, access to the modifications themselves (e.g. buying additional hardware, downloading and learning additional software, etc.) can be an additional barrier for users and ultimately decrease access to interactive technology such as VR. (Mott et al., 2019; Teófilo et al., 2018)

Another way current developers have attempted to address accessibility is through innovative alternatives - creating a new product designed specifically to meet ability-based needs and demands, or expand usability to a targeted audience. This approach allows inclusion to be the forethought of development, but possibly in competition to or in accessory to other accessible tech. (Mott et al., 2019; Teófilo et al., 2018) Building accessibility into the initial design allows a [physically] disabled user to immediately engage with an interactive experience and have their needs met outright. For example, integrating VR headsets to be built directly into someone's glasses would allow users with specific vision needs to engage with VR without having to remove their glasses. (Mott et al., 2019) These innovations could help accommodate ability-based barriers to using VR hardware such as: head and shoulder strength required to use HMDs, gripping controllers with missing digits, or low hearing. (Mott et al., 2019; Hamilton, 2018) Conversely, in accommodating one sector of ability, others may be excluded or neglected (eg: blind users vs. deaf users). (Teófilo et al., 2018; Mott et al., 2019) Given that physical disability is complex and varies from person to person, this approach can be inefficient.

Lastly, accessibility can be achieved through expanding development - utilizing and building features into existing software and systems. Current research in accessibility and interactive tech seems focused on creating new products or producing new redesigns of existing ones to achieve accessibility. However, there seems to be less focus in utilizing or further expanding features from existing tech, which could be an efficient starting point in addressing accessibility in VR. As VR software improves and becomes more sophisticated, there are emerging opportunities to improve accessibility that do not require creating something new (Wobbrock et al., 2011). Developing features in existing VR software, in tandem with modification and innovation, will ultimately open VR to a wider range of user abilities and audiences.

Motions Required for VR Games

There is little proper advertisement of types of motions required to enjoy and participate in various VR games. In the Oculus and Steam digital game stores, a game will usually only provide three indications of their physical movement requirements by way of hardware/play requirements: controller requirements (keyboard and mouse, gamepad, or motion controllers), play posture requirements (standing or seated), and play space requirements (room-scale tracking or in-place tracking). Games that require fast crouching to avoid an attack and a museum experience where one could walk around the exhibits would be indistinguishable via these

categories as they would have the same three labels: motion controllers, standing, and room-scale tracking.

This lack of detail in a game's categorical description forces one to use promotional materials to make guesses on their own to understand what would be physically required of a user. This makes mass examinations of how different games require a player to move difficult without actually experiencing a game for one's self. As such, I will describe a few broad categories of trends in VR gaming with specific detailed examples of games, their motions, and physical accessibility options within those categories.

1. Musical/ Rhythm Games

   In this category are games that feature a player surrounded by a virtual environment that must be reacted to in time with representations of the audio heard by the player. Usually, the player is able to stand in one place with minimal physical locomotion required as the environment scrolls towards them or away from them in some manner with objects to interact with. In *Beat Saber*, blocks that must be cut with the titular sabers scroll at a player, needing them to swing their arms in a specific direction, in time, to cut the block and score points. *Pistol Whip* takes this idea of targets that must be interacted with in time and makes them human shaped enemies that must be shot with a handgun to evoke an action movie sensibility. *Thumper* has a VR mode, but is played seated with a gamepad. Additionally, there are usually barriers that must be physically avoided either by crouching, walking, or leaning to one side. In *Beat Saber* and *Pistol Whip*, these barriers can be removed at a scoring penalty or by being disallowed from competing on standard global leaderboards.
2. Arcade Combat/Shooting Simulators

   In this category are games that feature a player in an environment or levels populated by waves of enemies that must be either shot or struck in some way. *Space Pirate Trainer* and *Robo Recall* are games that require players to reach behind them, as if to pat themselves on the back, to reach additional weapons. *Superhot VR* and *Gorn* (melee only) require the player to maneuver around the arena to pick up and use weapons obtained by leaning and reaching out without any assistance to draw weapons closer to a person at a distance, affectionately referred to as "force pulling" after the ability in the *Star Wars* franchise. All examples require players to physically dodge or block attacks to continue progressing in the game.
3. Experience Simulators

   These games are far more relaxed in pace, leaving a user to explore and interact with their environment at their own pace. Owlchemy Labs' games *Job Simulator*

and *Rick and Morty: Virtual Rick-ality* have optional time based challenges, but still require acts of crouching and leaning to physically reach locations where critical items are hidden to progress in the game. These games have limited options to work around this leaning and reaching action, focusing more on limiting locomotion requirements to a room with specific points to teleport to. *Star Trek: Bridge Crew* is played seated with motion controllers to interact with virtual touch screens to cooperatively pilot a spaceship.

4. Action Adventure Games

   These games are more like traditional video games. Some are games that have been ported over to VR like Bethesda's *Skyrim VR* and *Fallout 4 VR*, that utilize a combination of teleportation around the environment, motions to engage in combat similar to the combat simulators mentioned above, and traditional video game inputs via buttons. The VR video game industry has had two major releases come out in December 2019 and March 2020 in these categories that marks a shift to environments that are almost entirely physically simulated by the physics engine: *Boneworks* and *Half-Life: Alyx, or HL:A*. Both are narrative games where the player progresses through levels, solving physical puzzles by retrieving items or stacking items, climbing ladders, and engaging in gunplay. The experience simulator problems are seen here in needing leaning and crouching for finding necessary items to progress. *HL:A* and *Boneworks* both feature "force pull" powers, but also require players to reach over their bodies to retrieve and store items on their bodies. *HL:A* features more accessibility options in allowing for teleportation for locomotion, teleporting past physical barriers requiring crouching, and other accommodations for comfort. *Boneworks,* however, actually eschews these options to achieve something closer to a one-to-one mapping of player motion and character motion. *Boneworks* also features a lot of climbing, performed by physically reaching over one's head over and over, and looking above one's head to progress in the game.

To summarize the motions required in most of these games briefly, many games require one to reach over their shoulders to their back to retrieve weapons that are then aimed or swung in front of the torso. Dodging is achieved by leaning the body to one side or the other, physically walking, or physically crouching. The environment is interacted with reaching to simulate picking items up off of tables and shelves from locations that are above or below the player, requiring more leaning or crouching. Additionally, all these games have different zones on the body used for holding items, so an accessibility solution must be able to be flexible enough to work with multiple games.

Proposed Solution

Inspired by controller remapping utilities and keyboard macros stringing together simple commands to do something complicated in existing games, the proposed solution aims to replay saved actions on a single button press. By capturing positional and rotational information of the controllers in a safe environment, it can be recreated one frame at a time by attaching that information to the in-game character's hands centered on the head position of the character. While the Unity demo constructed to demonstrate this relies on the controller's buttons to engage in the capture and replay features, it is trivial to set this system up on a timer and capture the controller inputs as well. This data is optionally written to a file to be saved or even shared with another system to build a library of recorded motions over time to get rid of the requirement of a one time motion performance.

The second stage would involve a combination of methods to keep controller motion in a comfortable range, a box drawn by the shoulders and hips, and evaluating which worked best. One possible method would be introducing a series of transfer functions to shrink the travel distance of the arm to perform the reaching and grabbing actions that might be required by a game to a less strenuous degree. Another possible method would be trying to solve the issues of games requiring players to reach awkwardly around their own bodies by creating a fence around this comfortable bounding box of motion previously mentioned and interpreting the act of leaving this fence as the action required.

Implementation

To demonstrate the idea of allowing users to record and replay gross movement for VR applications, I implemented an application in a Unity Game Engine environment, utilizing the *Oculus Integration* assets to easily translate between the user's real-world actions and the state of the game environment. A controller mapping was constructed for the Oculus Rift S's Touch controllers such that holding the B button down would record the positional and rotational data of the controllers for as long as the button is pressed. Pressing the A button begin the replay of the saved motion, where a saved list of positional Vector3 data and rotational quaternion data would be supplied to the user's avatar model and an in-place representation of the user's actions using 3 boxes as shown in Figure 1: black representing the HMD, red representing the left controller, and blue representing the right controller. This rudimentary box representation dynamically adjusts its height and hand position at runtime to match the user's. Pressing the X button will clear the saved data.

The environment also has a mirror the user can see their avatar, a Unity provided character that was rigged from the waist up. All parts of the avatar character will follow the user's motion via an inverse kinematics skeleton animating the character dynamically. This was done with Unity's in-development preview of their *Animation Rigging* package.

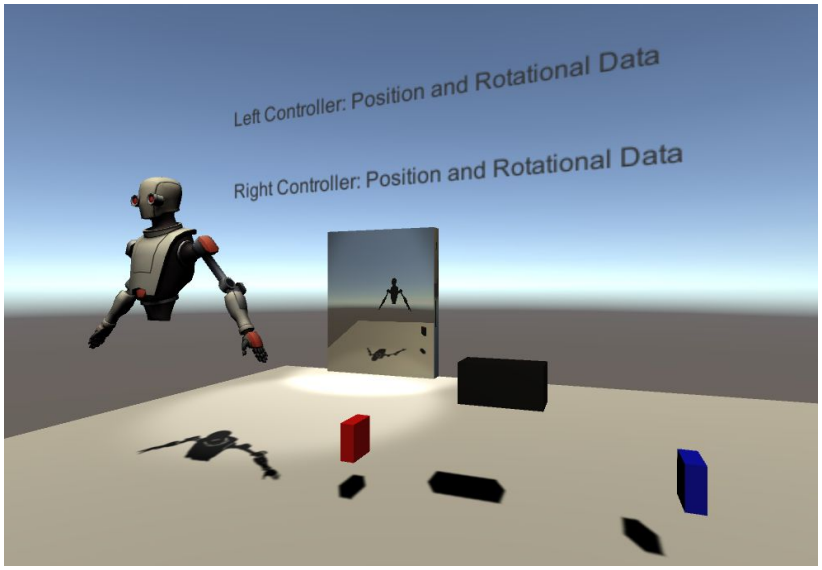

Figure 1: The basic environment with avatar, diagnostic placeholder text, and the rudimentary box avatar

Results and Discussion

Results

The Unity based demonstration performs admirably. Perfect fidelity is maintained between the capture and the reproduction of motion. While the character is briefly decoupled from the user's own motions at the same time as the replay, I did not find this a jarring experience as I initiated the decoupling, rather than one being forced upon me via a physics engine issue or a latency issue between myself and the headset's display. Introducing a ghostly marker of the user's true position while the replay occurs also helps the user keep track of their own frame of reference and motion. In my own personal experience, recording motion allowed me to overcome one of my issues common to VR games: storing important game items needed in an in-game fight exclusively on the back like *Boneworks*. Recording one proper reach over the shoulders to the back in an environment without time pressure created a consistent and accurate motion for me to use in place of needing two or three attempts to be able to reach a location. Outside of physical strain, this type of motion is complicated by the Oculus Rift S's camera tracking, which makes low-accuracy estimations of motion based on accelerometer and gyroscope data when the camera cannot see the controllers behind a player's back.

The second component of implementing transfer functions to reduce travel distance of a user's arm when reaching, while not successfully completed by a single undergraduate, was successfully studied by three researchers from the University of Waterloo and University of British Columbia and presented at CHI '20 (Wentzel, d'Eon, & Vogel, 2020). Their work does indicate that there is merit to this line of inquiry in trying to reduce the scale of motion required and that it can be performed comfortably and simply. Given more time, working to get a

successful implementation of my own and see if their results could be reproduced would be a high priority.

Limitations

Simply put, COVID-19 has introduced complications that make certain aspects of this project impossible. Testing with multiple individuals with equipment that must be strapped to one's face is too dangerous and prohibitively expensive to perform remotely. Additionally, a combination of technical issues and a lack of time due to health complications has kept the second stage involving the transfer functions and fence from coming being implemented in a working manner alongside the motion recording stage. The Unity demo created is actually a recreation that strips the functional replay environment from the non-functional combination environment.

While the motion replay technique is useful to overcome challenges while playing games, it still requires a one-time motion performance, which can still be challenging for those with any limits to their range of motion. Implementing the transfer function portion of the project would reduce the physical area needed to travel in the one-time recording in the instances where another person's stored and shared motion is not sufficient to overcome the challenge the user faces.

Improvements

Given more time, I would closely examine WalkinVR, a driver that uses SteamVR to modify user positioning and functionality for wheelchair accessibility and one-handed interactions. The ultimate goal with this undergraduate project would be to develop a similar utility for replaying and modifying motions in a similar way. Ideally, the need to personally capture this data in the first place could be replaced by a traditional piece of software that would produce the motion data from an animatable inverse kinematics skeleton, ensuring that anyone of any ability level could configure and play VR games independently.

References

Beat Saber [Computer Software]. (2018). Beat Games: Beat Games.

Boneworks [Computer Software]. (2019). Stress Level Zero; Stress Level Zero.

The Elder Scrolls V: Skyrim VR [Computer Software]. (2018). Bethesda Games Studio: Bethesda Softworks.

Fallout 4 VR [Computer Software]. (2017). Bethesda Games Studio: Bethesda Softworks.

Gorn [Computer Software]. (2017). Free Lives: Devolver Digital.

Graham, P. (2017). WalkinVR Showcases its VR Accessibility Software with new Videos. https://www.vrfocus.com/2017/07/walkinvr-showcases-its-vr-accessibility-software-with-new-videos/

Half-Life: Alyx. [Computer Software]. (2020). Valve: Valve.

Hamilton, I. (2018). A Practitioner Reflection on Accessibility in Virtual Reality Environments. *The Computer Games Journal*, *7*(2), 63–74. doi: 10.1007/s40869-018-0061-z

Job Simulator [Computer Software]. (2016). Owlchemy Labs: Owlchemy Labs.

Mott, M., Cutrell, E., Franco, M. G., Holz, C., Ofek, E., Stoakley, R., & Morris, M. R. (2019). Accessible by Design: An Opportunity for Virtual Reality. *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. doi: 10.1109/ismar-adjunct.2019.00122

Pistol Whip[Computer Software]. (2019). Cloudhead Games Ltd: Cloudhead Games Ltd.

Rick and Morty: Virtual Rick-ality[Computer Software]. (2017). Owlchemy Labs: Adult Swim Games.

Robo Recall[Computer Software]. (2017). Epic Games: Epic Games.

Space Pirate Trainer [Computer Software]. (2017). I-Illusions: I-Illusions.

Star Trek: Bridge Crew [Computer Software]. (2017). Ubisoft: Ubisoft.

SUPERHOT VR [Computer Software]. (2017). SUPERHOT Team: SUPERHOT Team.

Teofilo, M., Lucena, V. F., Nascimento, J., Miyagawa, T., & Maciel, F. (2018). Evaluating accessibility features designed for virtual reality context. *2018 IEEE International Conference on Consumer Electronics (ICCE)*. doi: 10.1109/icce.2018.8326167

Thumper [Computer Software]. (2016). Drool: Drool.

Wentzel, J., d'Eon, G., & Vogel, D. (2020). Improving Virtual Reality Ergonomics Through Reach-Bounded Non-Linear Input Amplifcation. *ACM CHI Conference on Human Factors in Computing Systems (CHI '20)*. doi: 10.1145/3313831.3376687

Wobbrock, J. O., Kane, S. K., Gajos, K. Z., Harada, S., & Froehlich, J. (2011). Ability-Based Design. *ACM Transactions on Accessible Computing*, *3*(3), 1–27. doi: 10.1145/1952383.1952384