Academic year
2021 - 2022

Open-World Virtual Reality
How Noticeable is Redirected Walking?

Thomas Van Onsem

Supervisor
Prof. Dr. Jeroen Famaey, IDLab, UAntwerpen

Dr. Filip Lemic, IDLab, UAntwerpen
Jakob Struye, IDLab, UAntwerpen

University of Antwerp
Faculty of Science

Disclaimer

This document is in accordance with the master thesis regulations and the Code of Conduct. It has been reviewed by the supervisor and the attendant.

Abstract

1. English version

Since 2016, virtual reality (VR) became available for the masses and consequently the technology started booming. With its popularity also came the challenges of providing multi-user experiences along with achieving a constant full immersiveness of the users during their virtual experience. This full immersiveness often requires simulating natural human navigation through unbound virtual environments (VEs). Redirected walking (RDW) algorithms provide a solution to this problem by taking advantage of the limitation of human spatial perception by unknowingly steering the users to walk in circles and keeping them within a limited physical space. As of today, there are not many extensively tested RDW algorithms, let alone algorithms that work in environments with multiple users. That is why, with this thesis, we developed a modular framework that enabled easy testing of different RDW algorithms in single and multiple user environments. Additionally, some improvements were suggested to increase the performance of all RDW algorithms being deployed in similar scenarios. Several live user experiments were conducted to evaluate the viability of the improvements and measure the performance and noticeability of RDW algorithms when used with different VEs, room sizes and number of users. The data from those experiments are provided to be used in other research. From the results of the experiments conducted in this thesis, we could conclude that our improvements caused an increase in overall performance and a decrease in noticeability for scenarios with both one and two users in different environment sizes. In addition, our framework proved useful for quickly and robustly testing the performance and user experience of different VR experience configurations.

2. Dutch version

In 2016 werd virtual reality (VR) geïntroduceerd bij de gewone consument, waarna de populariteit sterk steeg. Tegelijkertijd ontstond ook de vraag naar multi-user ervaringen, waarbij de gebruikers permanent ondergedompeld blijven tijdens hun virtuele ervaring met elkaar. Om deze volledige "immersiviteit" te behouden, moet de gebruiker op een natuurlijke manier genavigeerd worden doorheen een oneindige virtuele omgeving. Redirected walking (RDW) algoritmes bieden een oplossing voor dit probleem, door gebruik te maken van de limitaties van de ruimtelijke perceptie van de mens. Door de gebruikers onbewust in cirkels te laten lopen, slaagt RDW erin om hen onbewust binnen een beperkte fysieke ruimte te houden. Tot op heden werd slechts weinig onderzoek gedaan naar de performantie van deze RDW algoritmes, laat staan in omgevingen met meerdere gebruikers. Het doel van deze thesis was om een modulair programma te ontwikkelen waarmee verschillende RDW algoritmes eenvoudig getest kunnen worden, en dit in verschillende omgevingen met één of meerdere gebruikers. Daarnaast werden enkele verbeteringsmaatregelen ontwikkeld om de prestaties te verhogen van RDW algoritmes, gebruikt in deze context. Bovendien hebben we verschillende experimenten uitgevoerd om de haalbaarheid van deze verbeteringen na te gaan, alsook om de performantie en mate van "immerisiviteit" van RDW algoritmes te onderzoeken. Hierbij werd gebruik gemaakt van verschillende virtuele scenario's, met variaties in kamergrootte en in het aantal gebruikers. Data van deze experimenten zijn opgenomen in deze masterproef en kunnen van nut zijn voor toekomstig onderzoek. We konden concluderen dat, met het door ons ontwikkelde programma, performantie en gebruikerservaring van verschillende VR configuraties op een snelle en robuuste wijze getest kunnen worden. Bovendien bleken de voorgestelde verbeteringsmaatregelen veelbelovend, aangezien deze een positieve impact hadden op zowel de algemene performantie als op de "immersiviteit", en dit voor verschillende scenario's met zowel één als twee gebruikers.

Acknowledgements

There are many people I would like to thank for their help and support during the construction of this thesis. First of all, I would like to thank my promotor Prof. Dr. Jeroen Famaey for giving me the opportunity to perform this research and introducing me to the world of VR RDW. Next, I wish to express my deepest appreciation to my co-promotor Dr. Filip Lemic and supervisor Jakob Struye for their guidance, insights, patience and extensive feedback on this manuscript. Finally, I want to thank my partner, parents, brothers and friends for their support during the making of this thesis.

Abbreviations

2AFC	Two-alternative Forced Choice
3DoF	Three Degrees of Freedom
6DoF	Six Degrees of Freedom
APF	Artificial Potential Field
APF-RDW	Artificial Potential Field Redirected Walking
APF-R	Artificial Potential Field Resetting
HMD	Head-Mounted Device
IoT	Internet of Things
Pub-Sub	Publish-Subscribe
RDW	Redirected Walking
S2C	Steer-To-Center
S2MT	Steer-To-Multiple-Targets
S2O	Steer-To-Orbit
SRL	Steering algorithm learned via Reinforcement Learning
SSQ	Simulator Sickness Questionnaire
VE	Virtual Environment
VR	Virtual Reality
UDP	User Datagram Protocol

List of Tables

List of Figures

Contents

1. Introduction

In 2016, with the release of the HTC Vive, the Oculus Rift and the Playstation VR, virtual reality (VR) became available for the masses, marking it as "the year of VR" [1][2][3][4]. Henceforth, the industry started booming with sales increasing by 92.1 % in 2021. With a current market value of $11.64 billion and a projected value of $227.34 billion by 2029, VR is here to stay [5]. Entertainment, medicine, culture, education and many other fields have already taken advantage of this technology [6][7][8][9]. From walking through a city to the dissection of a muscle, VR allows us to cross boundaries that would otherwise be unimaginable. But, as with every new technology, it comes with challenges that need to be solved for achieving its full vision [10][11].

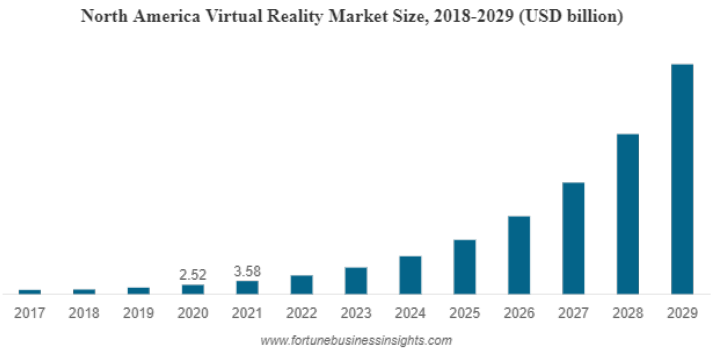


Figure 1.1.: Current and estimated market size of VR in the US [5].

One of these challenges focuses on achieving a constant full immersiveness of the VR users during their virtual experience. This full immersiveness often requires simulating natural human navigation through unbound virtual environments (VEs). Over the years, hardware solutions such as treadmills and foot platforms were provided as a solution to this problem. However, they come with a high cost, a high chance of motion sickness and limited portability [12][13]. As such, these types of systems have not seen widespread use and rarely make it to the average users.

As it stands today, the world of head-mounted devices (HMDs) can be divided into two categories: tethered VR and standalone VR. Each category has its own pros and cons. Tethered VR mostly gets its computational power from an external device, consequently having high visual

quality with rich graphics, as well as a consistently high frame rate. The downside is that the HMD is tethered to that external device, confining the users in their movement capabilities and gravely increasing the noticeability [14][15]. This limitation has been circumvented by strapping a computationally capable laptop to the user in the form of a backpack [16]. Even so, it is far from ideal: the user has to carry a weight on their back and the laptop's performance may degrade due to thermal throttling while confined within that backpack. The heat and the noise of the laptop venting out that heat will cause discomfort to the user. Standalone HMDs on the other hand have a built-in processor and are mobile, enabling the user to go anywhere without relying on other devices. However, they often have relatively low-quality graphics and a lower refresh rate because of the computational limits [17][18].

Additionally, the HMDs can be divided into two more categories: the outdated ones that use three degrees of freedom (3DoF) and newest versions that enable six degrees of freedom (6DoF). With 3DoF, only rotational movement is tracked, meaning the HMD cannot distinguish between a user moving and a user sitting down, which makes it very restrictive. 6DoF on the other hand, is called "Full VR" because of the HMDs ability to track all the user's movement as can bee seen in Figure 1.2.

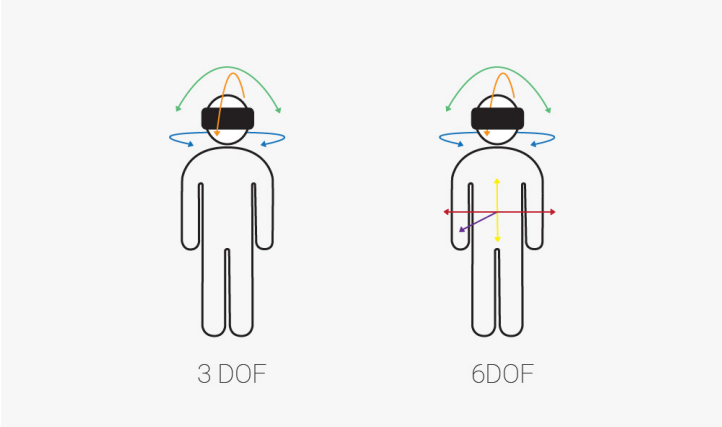


Figure 1.2.: Visualisation of the DoF principle [19]

Having a 6DoF non-tethered VR option gives us the opportunity to explore a cheaper solution to the immersiveness challenge. After all, other HMD types require the user to buy an expensive omnidirectional treadmill or computationally capable laptop [20][21]. Instead, we can look towards manipulation with algorithms: more specifically to Redirected Walking (RDW) algorithms. It is well-known that people have the tendency to walk in circles when wanting to walk straight in an environment without reference points, like a desert or dense forest [22]. RDW algorithms, initially developed by Sharif Razzaque, take advantage of this limitation of human spatial perception by unknowingly steering the users to walk in circles and keeping them within a certain limited physical space [23][24]. This results in the users being able to navigate through unbounded virtual environments while having the perception of walking

normally. This is typically accomplished in three ways:

1. **Curvature gain**: By rotating the virtual scene.
2. **Translational gain**: By increasing or decreasing the linear movement of the virtual camera resulting in a change of the length of the user's physical travel distance.
3. **Rotational gain**: By adding or subtracting extra rotation when the user is already turning.

This way the user will subconsciously veer along arced paths in the real world while walking along a different path in the virtual world, as can be seen in Figure 1.3. The RDW decisions will be based on context data produced by the HMD. More specifically: the user's physical location, steering direction and expected walking trajectory in the virtual world.

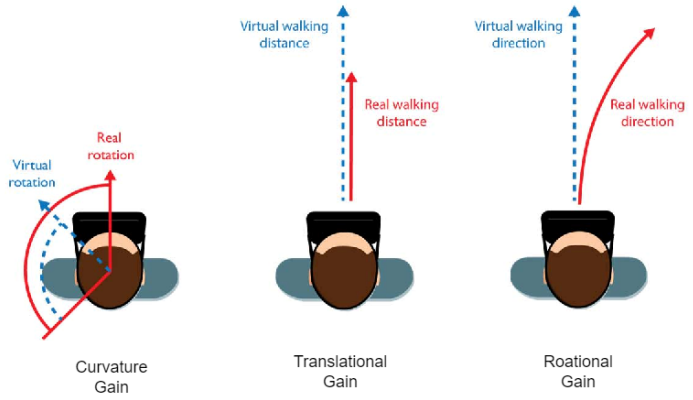


Figure 1.3.: Visualisation of the RDW principles [25].

Because the RDW technique is a rather unexplored method of improving the immersiveness of the VR users, it is rarely used by developers and companies at this time. Research is still being done to find the best RDW algorithm and to optimize immersiveness in all kinds of environments. For the most promising RDW algorithms, empirical data on both physical environment sizes and the user experience with regard to noticeability and nausea, is not abundantly available (see Section 2.2 for more details). It might be one of the reasons why the technology does not get picked up by companies developing VR experiences and software.

Together with the rise in popularity and the introduction of a global pandemic, another challenge emerged. The VR community and developers are not only interested in the immersiveness of the experience: social, collaborative, or even competitive VR gains traction in both home and commercial environments. With the announcement of Facebook creating the Metaverse on October 2021, the tone was set [26]. From then on, multi-user VR became one of the main focuses for Facebook, who, as a part of the Metaverse, wants to create the next version of the internet. A new way in which we would socialise, shop, play and interact with the world around us, partially accomplished with multi-user VR.

Not only that, but multi-user VR can impact the business experiences as well: meetings and classes can be virtualized, building environments can be constructed for real estate businesses and engineers, store an biolab layouts can be recreated, training spaces can be made to perform military exercises and healthcare training etc. [27][28].

Even so, multi-user VR experiences are very much a new technology and bring along some major complexities regarding immersiveness, interaction techniques, network capabilities, synchronization, security and adaptability of already existing virtual experiences and environments [29][30]. This adaptability also has to be applied to RDW algorithms, which were originally designed for a single user.

1.1. Research Question

This brings us to the question of how we can contribute to all of these issues. One can observe they all have the same scenario: one or more users, walking in a physically constrained space while being immersed in a virtual experience with a 6DoF enabled HMD. These users do not want to break out of their immersion to look for obstacles nor do they want to collide with them. To maintain this immersiveness, there is a need to understand the trade-offs between the physical environmental sizes and number of collocated VR users on the one hand, and the immersiveness, noticeability and experience of the users on the other. It brings us the following research question:

> How noticeable is redirected walking in virtual reality when taking into account different room sizes and number of users?

1.2. Our Contributions

In the previous part, we mentioned some challenges that lie ahead in the world of VR. More specifically, increasing the immersion of users in different environments and adapting VR technologies to enable multi-user experiences. To tackle these issues, we accomplished the following:

- *Develop a modular wireless framework to easily apply and analyse any RDW algorithm to any VE.* To the best of our knowledge, such a framework was not available yet. If the RDW algorithms were implemented at all, they were placed inside the game engine,

which complicates the multi-user setup and increases the computation for the HMD. By developing this framework, we aimed to provide researchers, developers and companies with the possibility to quickly setup, test and analyse the viability of any RDW algorithm and VE.

- *Develop a set of generic improvements applicable to all RDW algorithms and test those improvements on the most viable RDW algorithm.* As far as we know, most of the RDW algorithms have not been used with a standalone headset in a wireless multi-user environment. To make such scenarios possible and to optimize them, we introduced improvements that tackle positional drift, oscillating behaviour, latency, incorrect positional data and border hugging.

- *Test the noticeability of the most viable RDW algorithm in different VEs and with different number of users.* This was done to demonstrate the utility and modularity of the developed framework. Additionally, the practicality of certain configurations (room size, VE type, number of users) was determined and data about the experiments of those configurations was provided. These experimental data may be of use for the scientific community to easily test other approaches utilizing already collected traces from complex experimentation. The usefulness of our improvements was also confirmed and showed we could significantly reduce the noticeability and improve the overall experience of the user with certain room sizes.

1.3. Structure

The thesis is structured as follows: Starting with Chapter 2 we introduce the background concepts and related work. In Chapter 3, several generic improvements, applicable to a number of RDW algorithms, are suggested. Chapter 4 contains the design and development strategy of the framework as well as the setup of our experiments. Thereafter, in Chapter 5, we present the results of the performed experiments and discuss them. The thesis is then concluded in Chapter 6.

2. Background and Related Work

2.1. Redirected Walking Algorithms

Because the RDW technique has a lot of potential to increase the full immersiveness of VR users, a lot of researchers are delving into the world of RDW and started developing, improving and comparing algorithms to each other. While nearly all RDW algorithms use the same steering techniques (e.g., bending straight paths, altering user rotations, etc.), they differ on the high-level strategy of where a user is to be steered. These differences cause a variation in amounts of physical space required for the algorithms to function well, thus making some of them more effective than others.

2.1.1. "Steer-To" Algorithms

The "steer-to" algorithms, proposed by Razzaque et al., essentially try to steer the user towards a goal in the environment [23]. All three algorithms are rather basic but effective, thus often used as a baseline in comparison with newly developed RDW techniques. The three "steer-to" algorithms consist of:

- Steer-To-Center (S2C): The algorithm attempts to steer the user towards the center of the tracked space. This area is typically the position that is furthest from the walls and the location least likely to incur a future collision.
- Steer-To-Orbit (S2O): The algorithm attempts to steer the user in an orbit around the center of the tracked space. It is a way to keep the users in a fixed physical path, allowing them to endlessly roam in the VE.
- Steer-To-Multiple-Targets (S2MT): The algorithm attempts to steer the user towards a number of targets sequentially. It tries to achieve the same as S2C but additionally maximizes the utilized space.

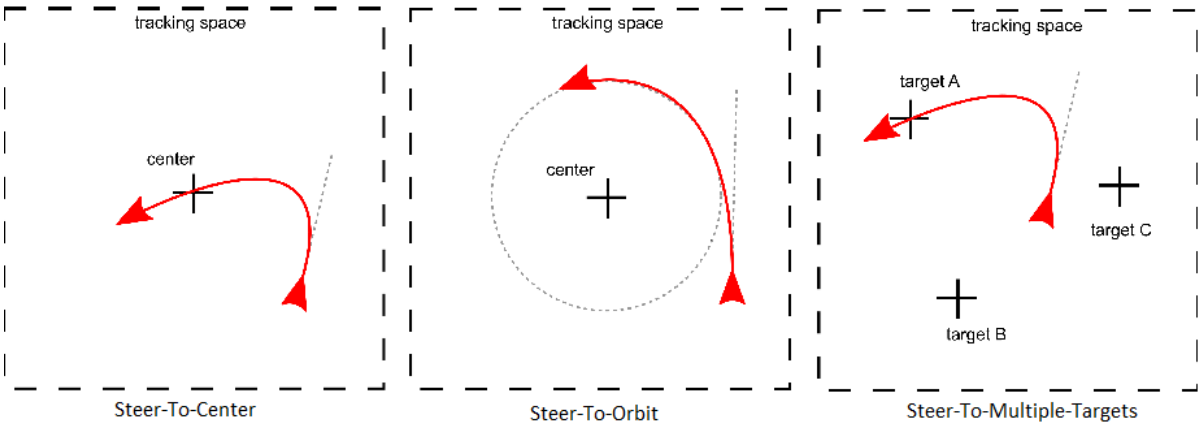


Figure 2.1.: Visualisation of the "steer-to" algorithms [23].

As can be seen, the algorithms are a basic form of RDW, designed to show the viability of the technique. They take very little context information from the headset and physical environment. Since they assume a collision-free environment at all times, they are also not suited for multi-user experiences or environments with irregular shapes and obstacles.

2.1.2. Steering algorithm learned via Reinforcement Learning

Recently, researchers have been looking at artificial intelligence to solve the problem of unknowingly steering the users to walk in circles and keeping them within a certain limited physical space. Strauss et al. attempted to accomplish this by applying reinforcement learning to the problem, calling it a Steering algorithm learned via Reinforcement Learning (SRL) [31].

The reinforcement learning algorithm consists of a learning agent and an environment. The agent interacts with an environment, receives both a reward and the next state from that environment and learns from both values. Eventually, through trial and error, it learns how to behave and knows what to do to achieve a certain goal [32]. This system is applied to RDW with the goal of maximizing the virtual distance traveled between possible collisions without adjusting the path of the user in a noticeable way.

The algorithm works in two steps:

- **The training phase**: First, a steering agent needs to be trained to work in the specific environment the user is going to walk in. This is done by creating a digital twin along with a virtual user that will walk trough that environment. The VR experience is then simulated by sending the user's physical position and orientation to the agent at certain time intervals. It then uses that data to choose the redirection gains that will be applied.

Based on feedback from these decisions, the agent's memory, containing which actions to take at what time, is updated. The agent is continually trained in this manner until the performance is deemed sufficient.

- **The application phase**: After training the agent, its memory, or more specifically, the parameters of the deep neural network that encodes the learned policy are frozen. The network may then be used to make steering decisions in new scenarios as a user continues to move around the VE.

The algorithm was compared to a scenario without redirection and one using S2C for 5000 virtual paths and 10 real user paths. They concluded that the policy learned on the simulated paths still significantly outperformed no redirection and had no significant difference from S2C. When trained with more than 10 real user paths, they expected the algorithm to perform better compared to S2C. Additionally, SRL has the ability to work in oddly shaped environments, which is not the case for S2C.

Even though SRL shows promise in the field of RDW, it is not yet suited for multi-user environments. This, because simulating one or more additional users, walking randomly in the same space, is quite the challenge. Consequently, the SRL agent cannot be trained to react in such scenarios. In the future, such scenarios might be possible. However, firstly more research needs to be done on SRL and multi-user experiences need to be configured and tested as well.

2.1.3. Artificial Potential Field

One of the more advanced additions to the collection of RDW algorithms, is the artificial potential field (APF) algorithm: initially proposed by Krogh & Khatlib [33][34] and adapted to be used for redirected walking by Hoffbauer and Bachman et al. [35][36].

For now, no RDW algorithm is advanced enough to imperceptibly redirect multiple users roaming in a physically restricted environment. Eventually, the users will collide with obstacles like walls or other users. To prevent this collision, the user will often get reset, i.e., they will be stopped and instructed to turn away from the obstacle [37][38]. The APF algorithm not only redirects users but also resets them when needed. That is why it consists of two elements: the RDW part, called APF redirected walking (APF-RDW) and the APF resetting mechanism (APF-R).

2.1.3.1. APF-RDW

APF-RDW takes a different approach in redirecting the users compared to S2C and S2O. Instead of attracting the users towards the center of the room or putting them in an orbit, APF-RDW repels the users away from obstacles and pushes them into open space. It is accomplished by generating a force vector that is directed away from obstacles and has a length inversely proportional to its distance from the obstacle. The steering direction is then determined by summing the individual force vectors, obtaining a total force vector that points towards open space, as can be seen in Figure 2.2. The force vector method significantly simplifies computa-

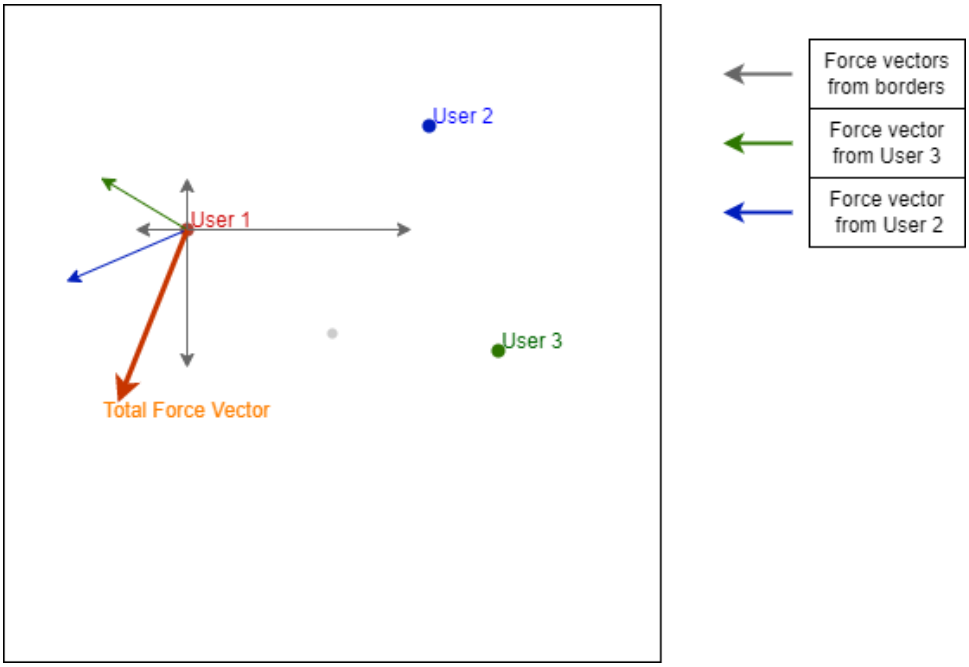


Figure 2.2.: Visualisation of the APF algorithm applied on User 1.

tional complexity compared to other algorithms: there is no need to predict collisions or make explicit decisions about how to prevent them. Subsequently, because the users constantly get repelled away from each other and from the closest point on a border, it also makes better use of the full tracking area as the users can disperse more fully into the space. Because of the way APF calculates its steering direction, it can easily be used in irregularly shaped physical environments. Additionally, the environment does not have to be collision-free: there can be numerous obstacles, both stationary and moving, like other users.

With live user tests, Bachman et al. showed that the APF-RDW algorithm improved upon S2C (for single users) in increasing the average distance between resets by 86 % and decreasing the number of resets by 64 % [36]. This, of course, majorly improves the immersiveness, decreases nausea and increases the overall experience of the users using an RDW technique.

Nonetheless, it still leaves room for improvement. As mentioned in the introduction, there are multiple ways of redirecting a user through a physical space, of which APF-RDW only uses one (by rotating the virtual scene). The algorithm also was not optimized for both performance (tweaking of force vector sizes, combine with other algorithms, add path prediction...) and some types of VR setup, as we will see in the next sections.

2.1.3.2. APF-R

As mentioned before, sometimes a collision is imminent and cannot be avoided. In this case, the resetting algorithm is triggered and the users get stopped in their tracks. APF-R then uses the total force vector, calculated similarly to APF-RDW, to determine the angle the user has to physically turn towards. To keep the break in immersion as small as possible, the previously mentioned curvature gain method is used to perform a 2:1 turn. In a 2:1 turn, the user's rotational speed gets increased, resulting in the user always turning 360° in the virtual world while in reality only turning a smaller computed angle in the physical world. This way, the user physically gets directed towards a safe zone while virtually having the exact same direction as before [39].

In their paper, Bachman et al. showed that with an environment of 25×45m at least 8 users could simultaneously roam through the space with a small number of resets [36].

2.1.4. Conclusion

With our different scenarios in mind, the ideal RDW algorithm would provide us with the possibility to properly redirect multiple users in differently sized environments while also performing better than others.

The "steer-to" algorithms were initially developed for single-user systems. Over the years, however, some additions to the algorithms were suggested, mainly to improve on the overall redirection angles and ability to work in irregularly shaped environments [40][41][42]. However, none of them solved the issue of multi-user experiences. That is why the "steer-to" algorithms were not used for the thesis experiments.

The same reasoning can be used for SRL. Although it is a promising way to redirect users, it is still very much untested and multi-user experiences are not yet possible. Additionally, the limited duration of this thesis would not allow us to collect the large amounts of data needed to train the algorithm. Therefore, we did not use SRL in this thesis either.

Even though APF still is not perfect, we could conclude that it is the best candidate for our wireless multi-user setting with different room sizes and VEs. Consequently, we chose APF as the RDW algorithm to test during our experiments.

2.2. Experimentation with Multi-user Full-immersive VR with RDW

2.2.1. Immeriveness

Over the years, research has been conducted on the noticeability thresholds of RDW. Most of this research was done by Steinicke et al., who not only determined the threshold for rotational gain but also for translational and curvature gains [43][44][45]. To determine these limitations, they conducted several experiments involving 18 human test subjects, with backgrounds ranging from students to professionals with expertise in computer science, mathematics, psychology and physics. They let the users walk through a 10×10m or 10×7m darkened room 4 times for 4 minutes and utilized a tethered HMD combined with a backpack-fitted laptop. This way an unlimited 6DoF experience could still be provided while having an older 3DoF tethered HMD and the computational power of the laptop itself.

The limits regarding rotational gains were collected by forcing the users to turn their head 90° to the left or right and letting them decide whether the physical rotation was greater or not greater than the visually simulated rotation. They concluded that physical rotations between 68° and 107° cannot be distinguished from a 90° rotation. For the translational gains, a different scenario was used: users had to virtually walk 5 meters forward and were asked if they had travelled a greater distance or not. As opposed to rotational gain, users were able to accurately distinguish between virtual and real translational movements. Lastly, the same was done for curvature gains, where instead of adding and removing forward gains, users were now veered towards the left or right by rotating the VE. They were asked whether the path was curved to the left or not. Results showed they could unknowingly veer the users along a circular arc with a radius between 16 and 24 meters [43][44][45]. Needless to say, these thresholds are of major importance when tweaking RDW algorithms to minimize noticeability.

2.2.2. Deployment Environments

Steinicke et al. not only measured the noticeability thresholds but also showed that the minimum room dimensions lay between 35 m and 50 m, depending on the veering radius and consequently the RDW algorithm that is used. Hence, these large dimensions were mostly used during experiments described in related papers [46][41][36]. Researchers, collecting similar data, often used a 25×45m room at the Miami University campus called "The Hive" [47]. The room was specifically adapted to provide easy testing of HMDs that were not made to enable 6DoF. Additionally, infrared sensors (from InterSense and WorldViz LLC) are present to measure rotational and translational change of users roaming through their environment. The sensors were installed because of the inability of some older HMDs to provide positional or rotational data.

In all previously mentioned experiments, researchers used Wi-Fi to establish communication during their experiments [43][44][46][41][36]. Most of the time, this wireless connection was needed to send contextual data (e.g. position and rotation), generated by the infrared sensors to the user's laptop, running the HMD. The contextual data was then used by the RDW algorithm, running on the laptop, to redirect the users. As mentioned in the previous section, the reason behind selecting this methodology lies in the inability or inaccuracy of some older 3DoF HMDs to provide positional or rotational data.

However, almost all of those experiments were conducted in single-user scenarios. Bachman et al., who conducted multi-user experiments to prove the usability of APF, used the same methodology to provide all users the positional data of the others. Otherwise, they would not be able to react to imminent collisions with each other.

2.2.3. Conclusion

We concluded that, because of the limited capabilities of the HMDs at that time, a lot of equipment and setup needed to be arranged. Not only did they need a laptop for every user to run both the RDW algorithm and the VE but they also needed a sensors to enable 6DoF and a wireless network to communicate that data.

Since we now have the Oculus Quest 2, a newly released standalone HMD supporting 6DoF with high refresh rate and increased processing power, we opted for a completely wireless environment [18]. Not only does it improve the full immersiveness of the users (by not having a laptop strapped to the back) but it also enables the use of more computationally expensive RDW algorithms, since all the computation could happen remotely.

The sensors were not needed since the Oculus can generate its own contextual data and communicate it to the central node. It enables us to choose any physical environment we desired, including a field outside.

3. Redirected Walking Enhancements

The combination of our unique testing scenarios and the newly released Oculus Quest 2 headset that we would be using during our experiments, gave us the opportunity to design some generic improvements for RDW algorithms used in similar scenarios with similar setups.

3.1. Considered Scenario

We wanted to measure the noticeability of one or more users roaming through environments of different sizes while being redirected by a RDW algorithm. This meant we needed to constrain those users to a fixed physical space while enabling them to roam through an infinite VE. These fixed physical spaces had various sizes throughout the experiments to determine the difference in noticeability of being redirected in different room sizes. Additionally, we changed the number of users simultaneously roaming in the same physical space. Again, we did this to determine any change in noticeability when playing alone compared to playing with two or three users.

To enable all these configurations, a large enough space needed to be arranged to accommodate the largest environment size. This way we were able to perform all the experiments at one location. Smaller environments could then be virtually created, also giving the testers the ability to ”crash” into virtual walls instead of real ones.

A completely wireless environment was chosen to try to create optimal immersiveness. The multiple users, roaming through the environment, were being imperceivably redirected by a central entity, setup at the edge of the space, as seen in Figure 3.1. The central entity was envisioned to run the RDW algorithm, giving it a global view of the environment and all its entities, enabling those multi-user experiences. Based on the decisions made by the algorithm, the recommended RDW rotations were delivered to the users in a way that maximizes their imperceivable steering and minimizes the number of perceivable redirects.

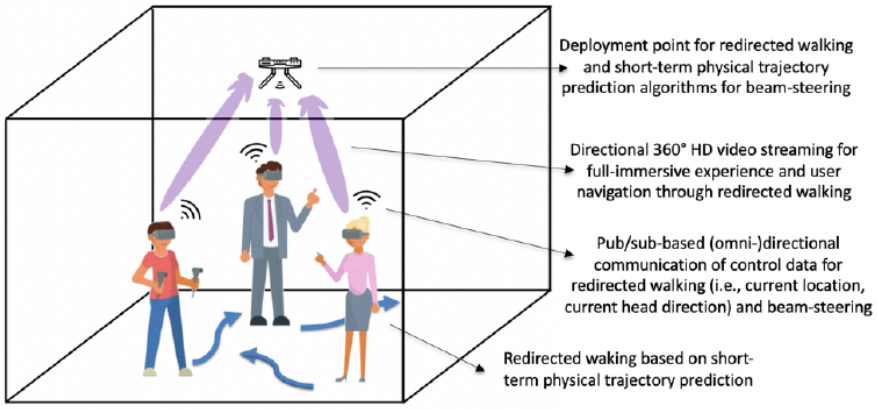


Figure 3.1.: The considered high-level scenario [48].

3.2. Design of RDW Enhancements

Our envisioned scenario was built as a multi-user environment with standalone HMDs wirelessly communicating with a base station running the RDW algorithm. This unique and untested scenario presented us with the opportunity to come up with some general improvements, applicable for every RDW algorithm in a similar scenario.

3.2.1. Latency Aware RDW

Because we worked with a wireless HMD, communicating with a central server to determine the steering recommendations, certain delays were to be expected. The end-to-end delay depended on both the speed of the RDW algorithm and the wireless infrastructure.

The RDW algorithms need a constant flow of context information to determine the best redirection angle. Consequently, a lot of traffic is to be expected during the execution of the virtual experience, making the delay measurements essential to optimize the algorithm and reduce the noticeability of redirecting the users.

Measuring the delays of wirelessly sending packets from the HMD to the server, allows us to extrapolate the real location of the user at the time of calculation, based on the received location, previous locations, rotation and speed. There are numerous ways to predict the real location of the user based on the latency. One of those ways is linear extrapolation, where we assume the user is walking in a straight line for the duration of a couple milliseconds. The server would first receive the positional data, run it trough the linear extrapolation and feed the new coordinates to the RDW algorithm, as seen in Figure 3.2.

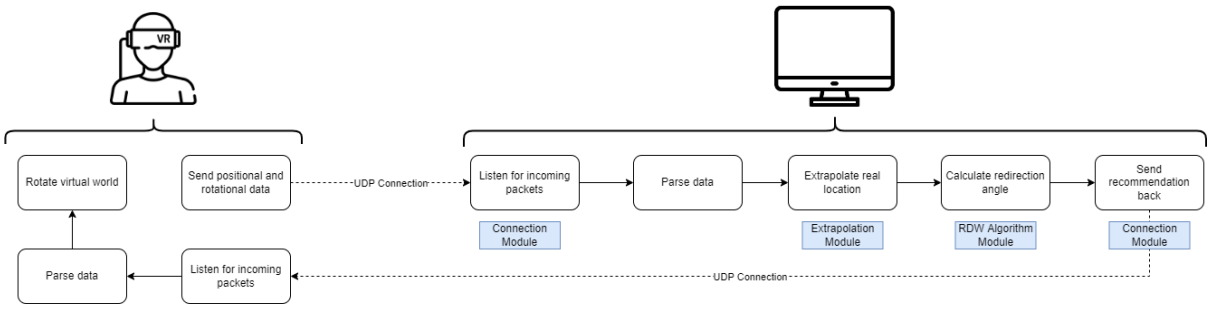


Figure 3.2.: Framework structure with latency awareness.

Since we investigated the noticeability of multi-user experiences, multiple protocols seem viable to use as communication between the HMDs and the server. One of the possibilities was to go for the fast but unreliable UDP protocol, that might collapse when used with multiple users. Other options included the complex but reliable distributed messaging protocols, that focus on frequently delivering data to multiple sources.

The latency aware RDW can also be applied to other RDW algorithms, receiving data indirectly and with a certain delay.

3.2.2. Ignoring Drift

It is well known that accelerometers and gyroscopes in HMDs are imperfect and have the tendency to provide noisy data [49]. This noise can propagate to other systems, used to locate and move the HMD and controllers, causing one or more of those components to drift. Most of the HMDs minimize this issue by using their on-device camera to locate fixed reference points in the play area. However, drift cannot be completely removed, meaning we need to account for it. To handle this drift, very small changes in positional data, received by the server, are discarded, as is shown in Figure 3.3. Determining this drifting threshold requires testing of the HMD when laying still and level. Slight changes in positional data then determine the threshold to recognize drifting.

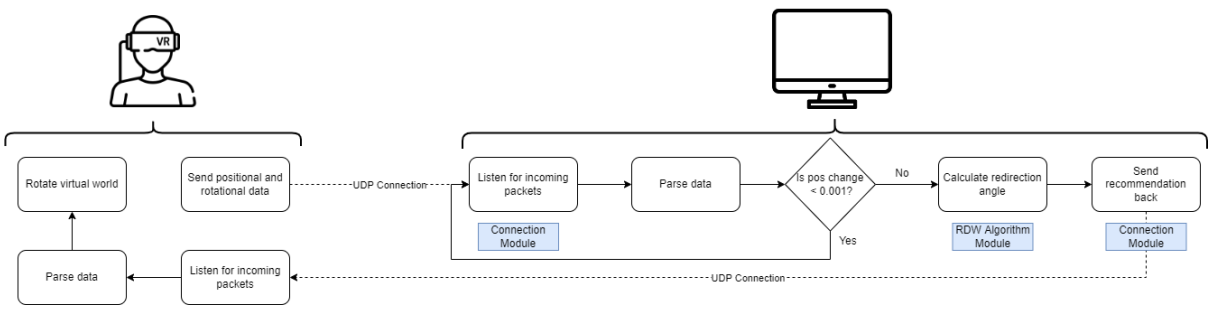


Figure 3.3.: Framework structure with latency awareness.

3.2.3. Ignoring Head Rotations

When talking about the position of a user, we actually mean the position of the HMD, which is mounted on a person's head. Consequently, a change in rotation of that person's head will result in a minimal change in position of the HMD and the RDW algorithm reacting to that position change. In some cases this can be desirable, especially for the algorithms using rotational gain to redirect users. However, in our case this change should be ignored since the user does not actually move and we only work with a rotation of the virtual world based on positional data.

Discarding this change is done similarly to our positional drift improvement. When positional data is received, it is matched with a predetermined threshold and either discarded or passed to the RDW algorithm (Figure 3.3).

Once more, this improvement can be used for every RDW algorithm using positional data from an HMD that moves when the user turns his head.

3.2.4. Reducing Oscillations

Preliminary tests showed that there were issues managing a high throughput of data. The algorithm recommended small alternating positive and negative rotations that resulted in an oscillating behaviour of the path [40], as can be seen in Figure 3.4.

To dampen this behaviour, a history of sent rotations was kept and compared with real-time recommended rotations. When a series of small alternating negative and positive rotations were advised, the alternating data would be added together to remove the oscillations and still recommend a change in orientation (Figure 3.5).

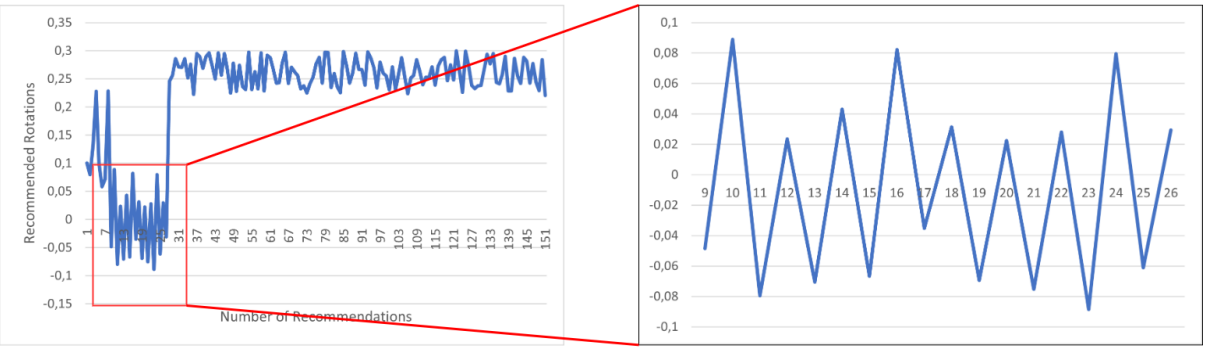


Figure 3.4.: Recommended steering angles from the APF algorithm during preliminary tests showing oscillating behaviour.

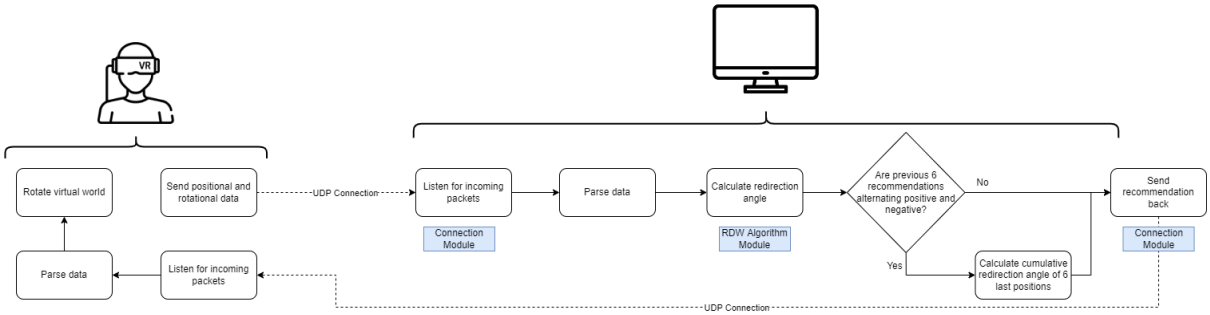
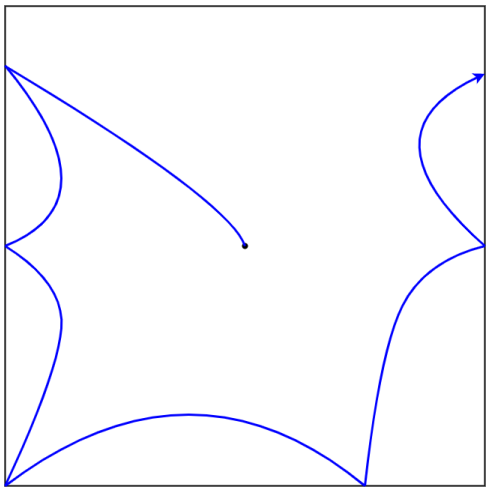


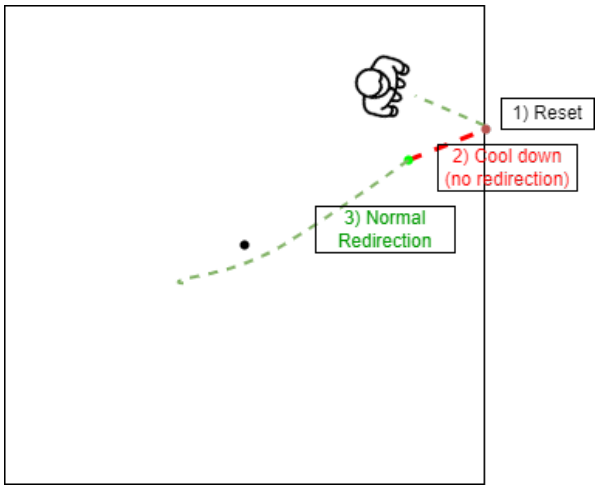
Figure 3.5.: Framework structure with reduction of oscillations.

3.2.5. Reducing Border Bouncing

The same preliminary tests also exposed a very specific and not yet documented problem in a multi-user scenario where a user would be sent into a reset loop. When 2 users were located close to both a wall and each other, the algorithm would reset the users towards the empty space and immediately redirect them back towards the wall, landing in a bouncing pattern with the walls, as can be seen in Figure 3.6a. It resulted in a very nauseating experience with a high number of resets.



(a) Abstract visualisation of bouncing behaviour in a specific multi-user scenario.



(b) Solution to border bouncing problem with cool down timer.

Figure 3.6.: Border bouncing problem and improvement solving the problem.

To tackle this issue, a cool down timer was implemented that would be initiated after a reset was triggered. This way, the user would be able to walk away from the wall and towards the empty space for a couple of seconds, thus maximizing the whole space and reducing the number of resets, as can be seen in Figure 3.6b.

This improvement could also be used as an addition to other algorithms not using the full

physical space and struggling with small bursts of resets.

3.3. Implementation and Optimization of Redirected Walking Enhancements

3.3.1. Enhancement: Latency Aware RDW

For one of our improvements, an accurate measurement of the latency between the HMD detecting the change of position and the access point receiving the information, needed to be measured. This latency highly depends on the protocol that is used to deliver the data to the server and back. Thus, it is important to test different options and choose the one that fits our system and has the lowest latency. Therefore, a small experiment was setup to, on one hand, test the detection speed of the HMD and, on the other hand, test the transmission delay of the protocols.

3.3.1.1. Setup

Four popular and applicable protocols were chosen to be tested as communication between the HMD and the server. Three of those protocols conform to the publish-subscribe (Pub-Sub) pattern that provides a structure for exchanging messages between publishers and subscribers. This pattern involves the publisher and the subscriber relying on a message broker that relays messages from the publisher to all the subscribers [50]. The publisher publishes messages to a channel that subscribers can then subscribe to. The following four protocols were selected:

1. **PahoMQTT**: A lightweight publish-subscribe mode messaging protocol regarded as a MQTT client implementation and designed for IoT applications in low-bandwidth and unstable network environments [51].

2. **RabbitMQ**: An open source message broker software that focuses on reliability, flexible routing and highly available message queues [52].

3. **Zero-MQ**: An asynchronous messaging library that does not need a broker and uses a message queue together with sockets that can represent a many-to-many connection between endpoints [53].

4. **UDP**: A datagram oriented protocol that uses a simple transmission method without hand-shaking dialogues for reliability or data integrity. UDP also tries to avoid the computing overhead of error checking and correction. It was chosen over TCP because of

its low latency. As mentioned before, the downside of UDP is the possibility of packets being lost. However, because of the high flow of context information being sent from the HMD, this was tolerated. It did mean we needed to make sure that all data, generated at the same timestamp, was being sent in the same message. This way, each message stayed independent of each other.

Now that we have chosen the protocols that would be used to send data over from the HMD to the server, a C++ program needed to be made to call those libraries and run natively on the HMD itself, using the VrApi provided by the Oculus development environment [54].

The program constantly checked a change in position, taking VR "drift" into account, and sent the current coordinates to the server using the previously mentioned protocols. When movement was detected, the screen turned green to enable us to capture any detection.

The HMD was mounted on a Rover Robotics 4WD[1][2] to more easily simulate and detect harsh movement. The server was setup on a laptop (MSI GS66) with a 120 Hz display as a Python 3.9 program that constantly checked and displayed received packets.

To film the sequence of events, we opted to record the experiment at 480 frames per second (fps) with a OnePlus 6T. It left an error of [0-2,083]ms. Since we needed to simultaneously film the HMD display turning green and the laptop showing a received package, the camera was setup on a tripod, as seen in Figure 3.7.

[1] Four-Wheel Drive
[2] https://roverrobotics.com/collections/all/products/4wd-rover-pro-variants

Figure 3.7.: Latency experiment setup

As mentioned before, the experiment consisted of the following actions:

1. Start server
2. Connect Oculus wirelessly to server
3. Start recording
4. Drive robot forward at full acceleration
5. Stop movement and recording once server prints received packages

This experiment was done 20 times on the X, Y and Z-axis to provide us with enough data and to possibly detect a difference in position polling on one of the axis.

Device	Framerate	Function
Oculus Quest 2	120 Hz	Detect movement, send packet and show green screen
Rover Robotics 4WD	/	Provide sudden movement
MSI GS66	240 Hz	Receive packet and print
OnePlus 6T	480 fps	Record both laptop and HMD

Table 3.1.: Latency experiment devices

3.3.1.2. Results

Wireless Communication Measurements were done by counting the number of frames between the HMD screen turning green and the server showing the incoming package. The delay in milliseconds could then be calculated from this data with the formula:

$$Delay = \frac{Frames}{480} * 1000$$

For UDP, the results showed an average delay of (214.90 ± 7.29) ms, as can be seen in Figure 3.8. The error margin was calculated by taking half of the time between frames of all the used devices. For the HMD this was 8.33 ms, the laptop has 4.17 ms between frames and the camera 2.08 ms.

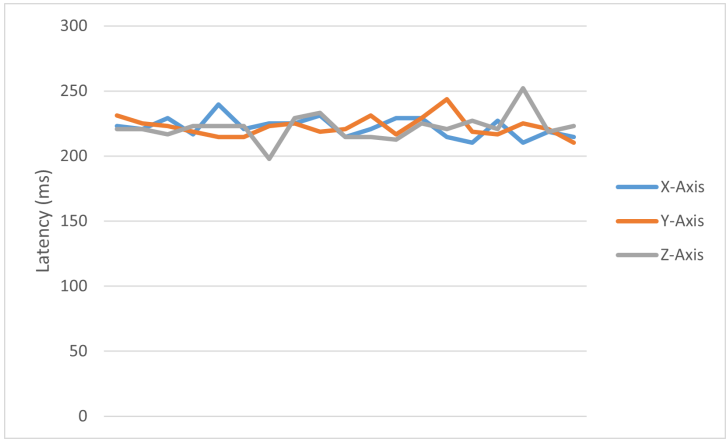


Figure 3.8.: Latency of UDP while sending packets between the HMD and the server.

The Pub-Sub libraries were also tested with their default settings and produced the following results [53][52][51]: For ZeroMQ, we measured a delay of (256.52 ± 11.46) ms while MQTT took (281.74 ± 11.46) ms to send a package from the publisher to the subscriber. RabbitMQ was the slowest at (316.93 ± 11.46) ms.

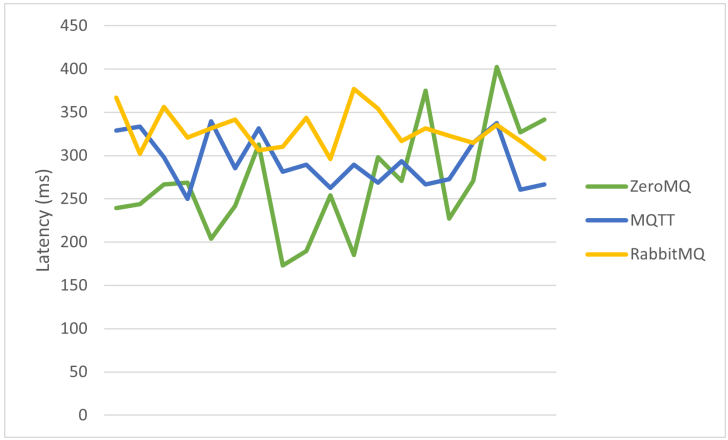


Figure 3.9.: Latency of the Pub-Sub mechanisms.

Because of the earlier research done on these systems, these results were expected. In literature, it was already described that ZeroMQ generally performed better compared to MQTT when used with smaller packets (which was the case) because of the absence of a broker. It resulted in the subscribers and publishers being connected in an end-to-end manner consequently enabling the subscribers to keep up with the publisher [55]. RabbitMQ also did not perform that well compared to ZeroMQ because of the batch processing it employed to improve throughput at the expense of latency performance [56].

We can conclude that our results showed a discrepancy between the four transmission systems. UDP seemed to outperform the three pub-sub libraries with at least 45.79 ms. We assume the discrepancy is caused due to the difference in quality of service the pub-sub mechanisms provide compared to UDP.

Movement Detection To give a full picture of the delay between the user moving and the server getting that movement info, we roughly measured the number of frames needed between the robot moving and the headset detecting that movement. Because it is hard to exactly determine at which frame the robot started moving, the results served as an indication and had a large error margin.

The measurements gave in an average delay of (50.23 ± 20.82) ms for the Z-axis (forwards and backwards), an equal (48.14 ± 20.82) ms for the X-axis (left and right) and a larger (92.55 ± 20.82) ms for the Y-axis (up and down), as can be seen in Figure 3.10.

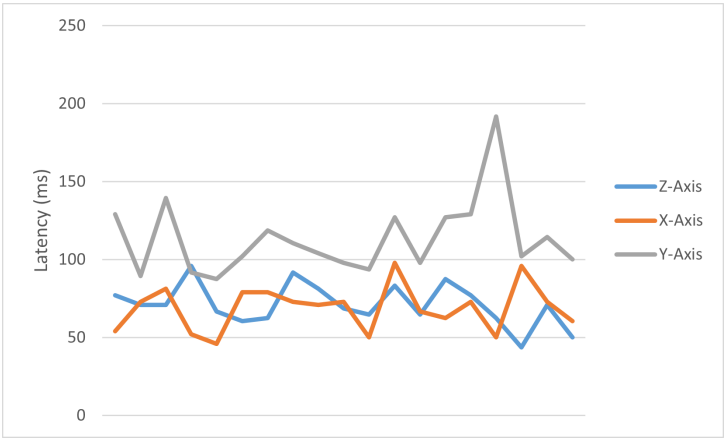


Figure 3.10.: Latency of change detection.

We noticed the Y-axis having a higher average delay in detecting movement compared to the other two axis. There are two hypotheses that can be put forth for this discrepancy. We believe it is either caused by the HMD trying to save power by infrequently polling a lesser used axis or

a wrongly calibrated sensor. Nonetheless, it was not an important element for our experiments, where we mainly used the X and Z-axis.

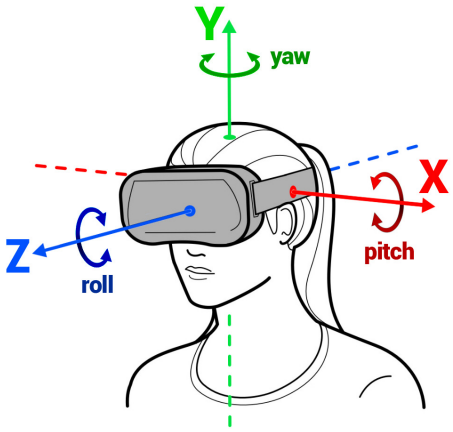


Figure 3.11.: The different axis of an HMD.

Conclusion The results regarding wireless communication prompted us to select UDP as communication protocol since we focus on latency instead of reliability. Due to the high throughput of data from the HMD to the server, packet loss could be tolerated. Future experiments, testing more than two users, might prove that UDP is not equipped to handle such a large volume. It might prompt us to instead choose the Pub-Sub based protocols that fit more naturally in the multi-user setup, making latency awareness all the more important.

Our data gave an indication on how long it took to generate context data, send that data to the central entity and sending back the recommended rotation:

$$Latency = \frac{50,23 + 48,14}{2} + (2 \times 214,90)$$

It resulted in using 478.98 ms as a baseline for our latency aware RDW. As we have seen before, this total delay is significant in a VR context, where motion sickness could easily be triggered. Therefore, introducing a latency aware RDW algorithm should certainly improve the performance of the RDW algorithm.

4. Experimentation Setup and Methodology

Conceptually the experiments consisted of the users walking in a, mostly unbounded, VE while being confined to a restricted physical environment. Positional data of these users was sent from the HMD to the laptop, where the RDW algorithm steered the users inside that restricted physical environment to avoid collisions with other users, borders and obstacles.

With this setup we measured the performance of the RDW algorithm for different types of environment sizes, different types of VEs and various numbers of users. Additionally, the usefulness of our improvements were evaluated as well.

4.1. Infrastructure and Setup

4.1.1. Framework

To our knowledge, no previous software was made to enable RDW in a wireless environment with multiple users. Consequently, a modular framework[1] needed to be developed to enable the various planned experiments.

The framework was built in C++17 and has 4 important parts, as can be seen in Figure 4.1:

- **The core of the framework**: it contains code to enable communication with the HMDs via a listening function. Additionally, it contains code to receive contextual data, code to parse the received data and convert everything into data structures and a system to manage the different users in the VE.
- **The RDW algorithm module**: because we wanted to make the system as modular as possible, we made it in a way that attaching different RDW algorithms does not need any

[1] https://github.com/ThomasVanOnsem/VR_RW_Noticeability_Public.git

change in the code of the framework itself. Because the HMD only needs an adjustment angle, a single parent function was created, called *predictRotation()*. When adding a different RDW algorithm, the developer can then simply override that function to attach the new algorithm to the system.

- **The extrapolation module**: we see the extrapolation of the user's actual position (in function of the delays) as a significant improvement to the overall RDW system. The extrapolation (currently implemented as linear extrapolation) can take many forms and can get as complicated as the developer wants (e.g.: polynomial or conic extrapolation, supervised learning etc.). That is why, in parallel with the algorithm module, we made this module easily interchangeable as well. The general function *extrapolate()* was constructed to open up the system to new extrapolation methods.

- **The tools module**: this module mainly consists of the system to log all the received and produced data along with some other tools that enable the developer to offset the users' position or invert their orientation. These adjustments are always made by calling a single function with the needed parameters.

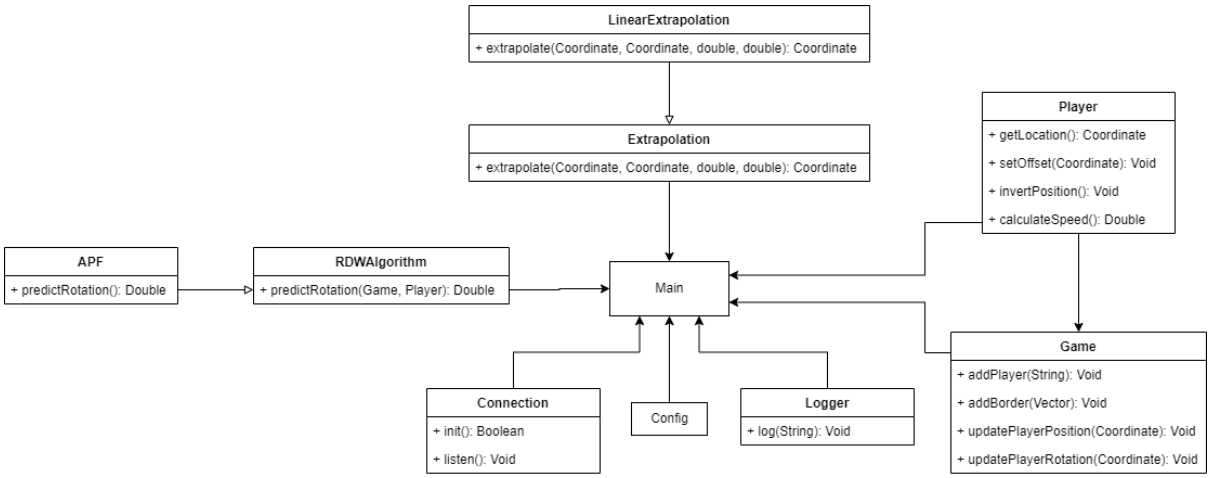


Figure 4.1.: Structure of the testing framework.

In addition to to the framework, the HMD also needed to send the required info wirelessly to the server. Because all the RDW computation was done server-side, the amount of code needed on the HMD to make everything work, was limited. The environments were built using Unity, which meant that the built-in C socket module could be used to send and receive the appropriate data via UDP. Other unity functionality also enabled the rotation of the VEs based on the received recommendations.

4.1.2. Algorithm

As mentioned before, there are not a lot of options when it comes to RDW algorithms. Ideally, we would test all of them with our improvements to more reliably prove their usefulness. But, because of the reasons mentioned before, we chose to focus on APF for our noticeability experiments.

4.2. Scenarios

4.2.1. Deployment Environment and Setup

The experiment was planned to both include commercial and non-commercial cases, meaning various room sizes were used to test the noticeability. More specifically, we went for a 5×5m, 10×10m and 15×15m environment size. We chose these sizes to determine the minimal environmental requirements of using RDW. Ideally, RDW would work in practically feasible room sizes, enabling everyone to utilize the system. Large environments like the Hive, mentioned in related work, are not abundantly available, consequently limiting the usefulness of RDW. Therefore, the possibilities of using RDW in manageable environment sizes were explored.

To accommodate all these environment sizes, we chose to look outside and select a 106×65m field (Fig.:4.2) near the Middelheim campus of the University of Antwerp. Physically, those room sizes were precisely measured and marked with cones. Virtually, we manually entered artificial coordinates marking the play space.

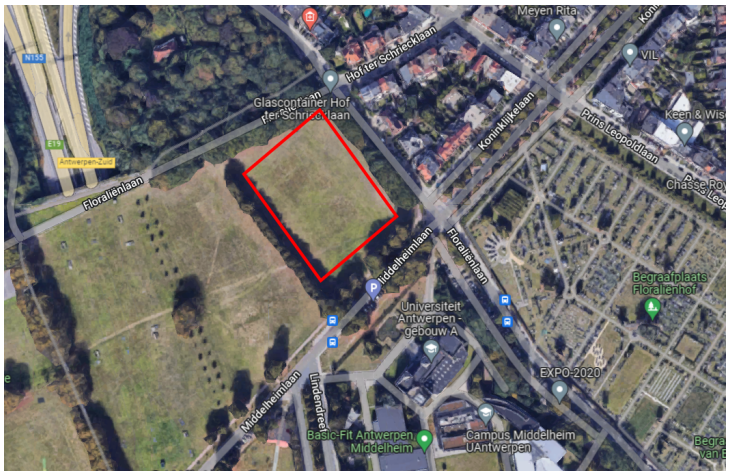


Figure 4.2.: Location of Noticeability Experiment

During the execution of our experiments, 3 devices were used:

- The access point: a wireless connection between the devices needed to be established to make everything work. This was accomplished by making a mobile hotspot using a Samsung Galaxy S8. Any other type of AP could, of course, reduce or increase the latency, depending on it's capabilities.

- The server running the RDW algorithm: A MSI GS66 laptop with an Intel i7 processor, 16GB of RAM and Wi-Fi 6, running Windows 10.

- The HMD(s): An Oculus Quest 2 with a Qualcomm Snapdragon XR, 120 Hz refresh rate and 6GB of RAM, running Android.

Performing such experiments outside presented us with a couple of challenges. Firstly, the Oculus controllers are identified by the HMD trough infrared beams, meaning the presence of direct sunlight prevents the HMD to localize the controllers, rendering them useless. This problem was circumvented by disabling the controllers and using the buttons integrated on the HMD. Secondly, operating an HMD in direct sunlight can cause major damages to the internal screens. This is because the large lenses, used to magnify the internal screens, can direct and cluster the sunlight onto the internal screen, causing significant damage. We solved this issue by only conducting the experiments on heavily clouded days and by attaching a small cover to the HMD, preventing any sunlight from landing in the lenses when not mounted to the head of a user. Lastly, we did not notice any distortion with the on-device tracking cameras caused by the direct sunlight.

4.2.2. Different Virtual Experiences

We designed two environments in Unity (v.2020/03) to test the noticeability of RDW algorithms:

- **Straight Path World** (Fig.:4.3): The user was forced to follow a straight path during the full duration of the virtual experience. This was considered a worst case scenario since the RDW algorithm will have the hardest time unnoticeably redirecting the user. Virtual experiences also rarely let a user walk in a straight path for minutes at a time.

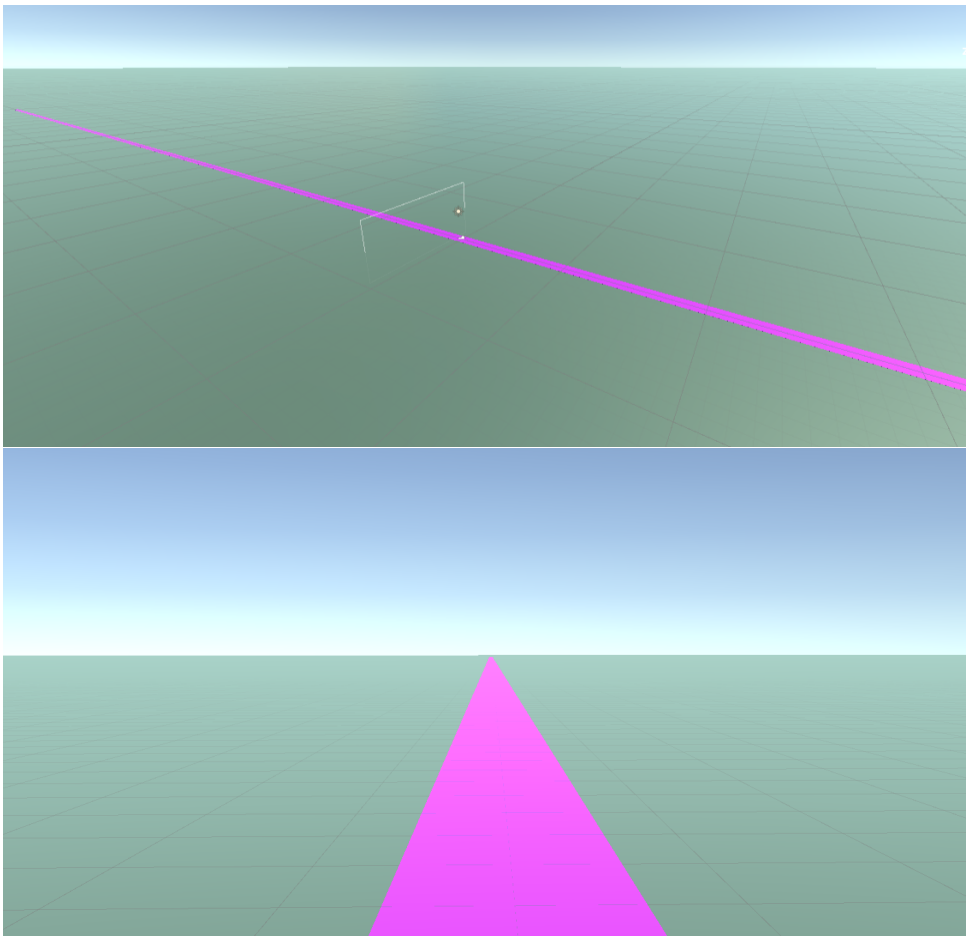


Figure 4.3.: Virtual Environment 1: Straight Path

- **Random Path World** (Fig.:4.4): A more realistic test where a user walked in an open environment and was encouraged to follow a curved valley. Because the users virtually walked on a curved path, the curvature introduced by the RDW algorithm was expected to be less noticeable.

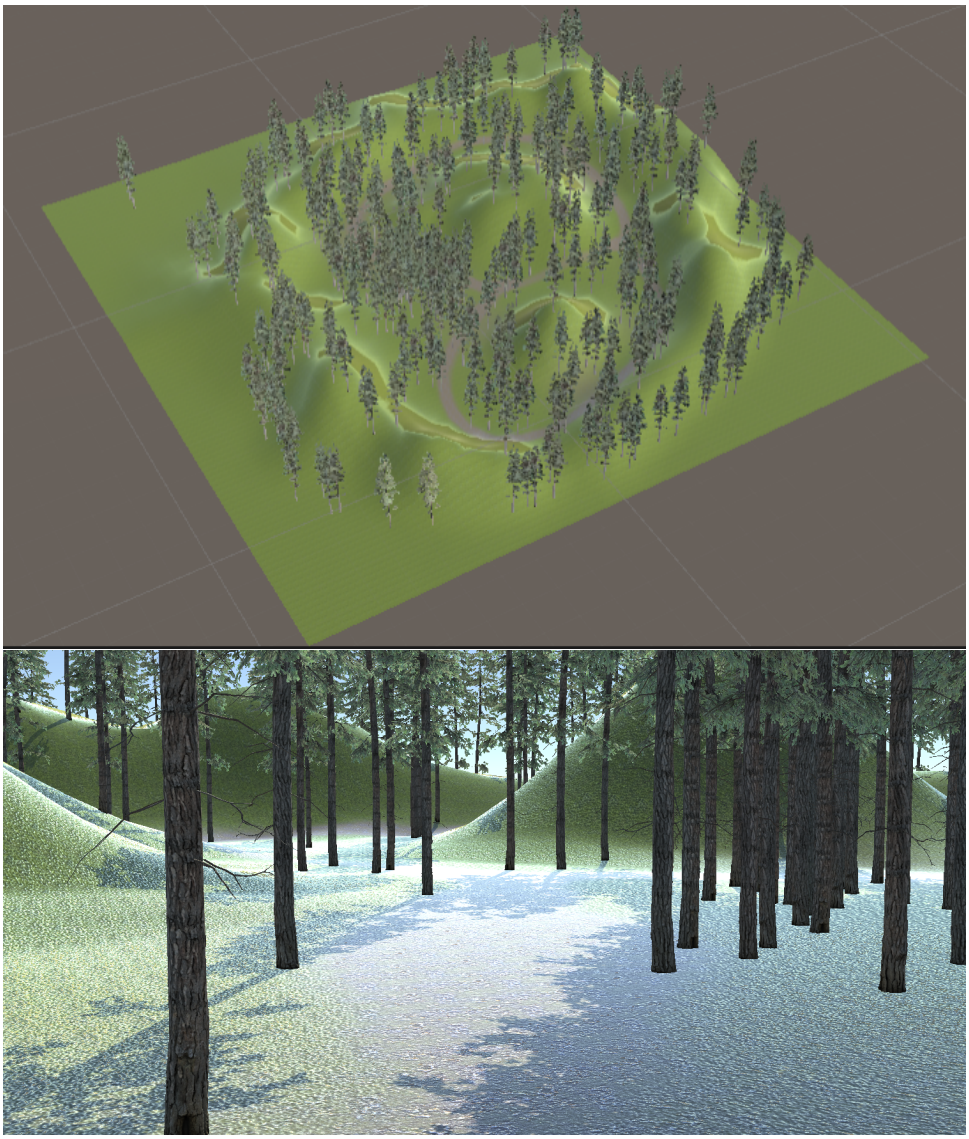


Figure 4.4.: Virtual Environment 2: Random Path

4.2.3. Virtual Reality Users

With the extensive time that was needed to run the noticeability experiments, we decided to work with 6 testers. All testers had different academic backgrounds and VR experience to simulate real world users: as seen in Table 4.1.

User	Age	Sex	Height (m)	VR Experience	Background
U1	25	F	1.66	No experience	Pharmaceuticals
U2	52	F	1.63	No experience	Law
U3	51	M	1.83	Almost no experience	Law
U5	23	M	1.80	Some experience	Medical
U6	21	M	1.86	Some experience	Medical
U7	24	M	1.86	No experience	IT

Table 4.1.: Background of the VR Users used in the experiments.

To avoid bias, testers had no knowledge of the parameters and goals of the experiments. At the start of every test, they were told to simply follow the path and try to enjoy the environment. When the APF algorithm was used, they were also informed that a reset could happen. The users would see a stop sign followed by the world turning and guiding them in a different direction.

Since results regarding noticeability could decline after the experiences a test subject had with the environment, we never let the subjects run more than 4 experiments and ordered the experiments from the expected least noticeable to most noticeable experience.

4.2.4. Duration

For determining the duration of the tests, a trade-off needed to be made between the testers losing their concentration and the real world scenario of the virtual experience being played for multiple minutes at the time. Looking at the number of experiments to be performed and the limited time frame of the thesis, we landed on an experiment duration of 5 minutes. Because there are no distractions and interactions in the virtual experience, the users would lose interest after several minutes and start to focus on the rotations of the world.

4.3. Performance Metrics

Implementing a RDW algorithm and designing a VE where the users can roam freely is not enough to test the performance of an RDW algorithm in different settings. Certain performance indicators need to be determined beforehand. We mentioned that the experiments will include one or more users, running through an environment where APF is applied. Hence, we can measure the performance of the RDW algorithm using those predetermined indicators along with the noticeability and nausea of the users.

4.3.1. Algorithm Performance

To measure the performance of a RDW algorithm and to compare it with others, previous research used a couple of default metrics that are applicable to almost every RDW algorithm that exists [57][46][41][36][42][58]:

- **Average distance (in** m**) between resets**: The distance between resets shows us how well the algorithm uses the available space and how hard it can veer the users away from the walls or other users.
- **Number of resets**: Similar to the previous metric, it shows us how well the algorithm uses the available space and is a possible indicator of how nauseous the users become (after a large number of resets) [59].
- **Average steering rate (in °/s)**: The steering rate is a very important metric to keep track of. It is an indirect indicator of the noticeability and nauseousness of the users.
- **Average physical distance (in** m**) from center**: This metric is mostly used in comparison with S2C and S2O, that focus on the user being a certain distance from the center.
- **Total distance travelled (in** m**)**
- **Physical walking path**
- **Virtual walking path**

All of the indicators combined, give an exhaustive view of what the user went trough during the experiment and how the RDW algorithm behaved. On top of that, we added another indicator: the maximum steering rate. The metric was recorded to link with the noticeability and to compare with the previous research done by Steinicke et al., where the thresholds for noticeability and nausea in VR using RDW were determined [43][44].

4.3.2. User Experience

The main goal in this thesis was to measure the noticeability of APF. Other papers, measuring the noticeability, used two different psychometric methods [43][44][60][61][62][63][64][65]:

- **Likert Scale**: The Likert scale "is a five point scale that is used to allow an individual to express how much they agree or disagree with a particular statement" [66]. It is easy to set up, ask and process but it is sensitive to different kinds of biases. Subjects can, for example, have the tendencies to change their answers in an attempt to be "good" test

subjects during analysis [67].

- **Two-Alternative Forced Choice** (2AFC): A test that is designed by Gustav Theodor Fechner to measure the sensitivity of a subject to input [68]. It assesses the noticeability of testers to certain actions by subjecting them to cases where the action is active and cases where it is not. Testers are then asked to identify the cases where the action was active and where it was not. This would indicate the subjects actually recognizing the action by correctly being able to identify the correct situations. In our case, it means that experiments need to run multiple times (without and with RDW).

Because of the limited scope and time frame of this thesis and the more frequent use in similar literature, we went with the Likert Scale. The lack of knowledge the subjects had about the purpose of the experiments and the absence of any positive, negative or socially acceptable answer on the scale, allowed us to reduce the bias that may be present when answering the Likert scale.

To measure the performance of the algorithms, we kept track of the number of resets that were needed during the virtual experience. Resets introduce a perceptible discrepancy in the movement, which can be distracting and lead to nausea. Not only that, but the level of camera changes that are made can also induce simulator sickness: a form of motion sickness caused by interaction with a VE [69][59]. That is why, as a secondary psychometric data point, we also measured the nauseousness of the users after each experiment. This was done using a 5-point Likert scale, which is similar to the simulator sickness questionnaire (SSQ) used in [36].

4.4. Data Collection

4.4.1. Testers

The main goal of this thesis was to measure the noticeability of RDW algorithms in real world scenarios. This meant, we needed to question the testers at the end of every test for both nausea and noticeability of being redirected. This was be done using the Likert scale from Figure 4.5.

Lastly, since the psychometric data could vary based on the tester's profile, they were also questioned about their VR experience, their academic background and their age.

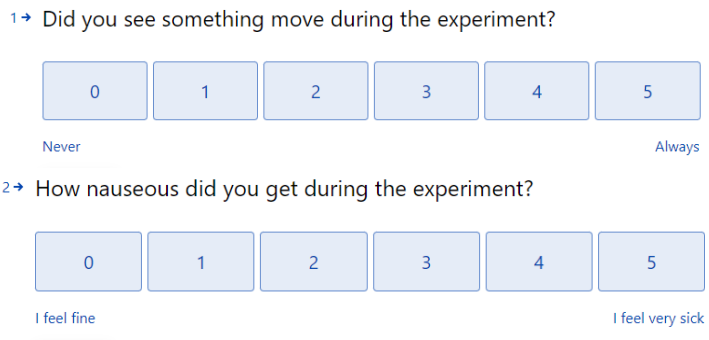


Figure 4.5.: Likert questionnaire used in the noticeability experiment.

4.4.2. Virtual Experience

In addition to to the questionnaire that was given to test subjects, all possible data was collected during the execution of the experiments. The user's position, rotation, number of collisions and the algorithm's recommendations were recorded to analyse the real world noticeability and performance using the metrics we defined in the previous section. Additionally, a timestamp was attached to every data point that was received or calculated during execution of the experiment.

While running, all the collected data was exported to a text file. First, a file was created containing the settings used to run the experiment and RDW algorithm. Second, a separate file was made for every user. It contained all the received and sent data from and to that user. This data consisted of the attached timestamp along with the type of data and the (x,y,z) coordinates for rotational and translational entries. Because we used the linear extrapolation and added the possibility to offset users, the log file included both the received translational data and the extrapolated and offset data. Recommended rotations were also printed with a timestamp, the type of data and a simple float.

```
14-05-2022_21-36-42 | Received virt pos: (-0.033972,1.134884,0.002432)
14-05-2022_21-36-42 | Received real pos: (-3.716028,-1.134884,3.747568)
14-05-2022_21-36-42 | Recommend Rotation: 0.000000
14-05-2022_21-36-42 | Received virt pos: (-0.033972,1.134884,0.002432)
14-05-2022_21-36-42 | Received real pos: (-3.716028,-1.134884,3.747568)
14-05-2022_21-36-42 | Recommend Rotation: 0.000000
```

(a) Example log file for a user.

```
14-05-2022_21-31-28 | Guardian: 0
14-05-2022_21-31-28 | Room size: 15
14-05-2022_21-31-28 | Use Delay: 0
14-05-2022_21-31-28 | Delay: 255.710007
14-05-2022_21-31-28 | # Players: 2
14-05-2022_21-31-28 | Border: (7.000000,0.000000,7.000000)
14-05-2022_21-31-28 | Border: (7.000000,0.000000,-7.000000)
14-05-2022_21-31-28 | Border: (-7.000000,0.000000,-7.000000)
14-05-2022_21-31-28 | Border: (-7.000000,0.000000,7.000000)
14-05-2022_21-31-28 | Offsets: (-3.750000,0.000000,3.750000)
14-05-2022_21-31-28 | Offsets: (3.750000,0.000000,-3.750000)
```

(b) Example log file for the environment.

Figure 4.6.: Example log files produced during the experiments.

4.5. Summary

All these variables combined result in a large amount of possible experiments to perform. The limited timeline forced us to select (Table 4.2) the experiments that were most promising, useful and realistic:

Virtual Experience	Physical Environment Size	Use Improvements	Number of Users
Straight	15×15	Yes	1
Straight	15×15	No	1
Straight	10×10	Yes	1
Straight	10×10	No	1
Straight	5×5	Yes	1
Straight	5×5	No	1
Straight	15×15	Yes	2
Straight	15×15	No	2
Straight	10×10	Yes	2
Random	15×15	Yes	1
Random	15×15	No	1
Random	10×10	Yes	1
Random	10×10	No	1
Random	5×5	Yes	1
Random	5×5	No	1
Random	15×15	Yes	2
Random	15×15	No	2
Random	10×10	Yes	2

Table 4.2.: Summary of the performed noticeability experiments.

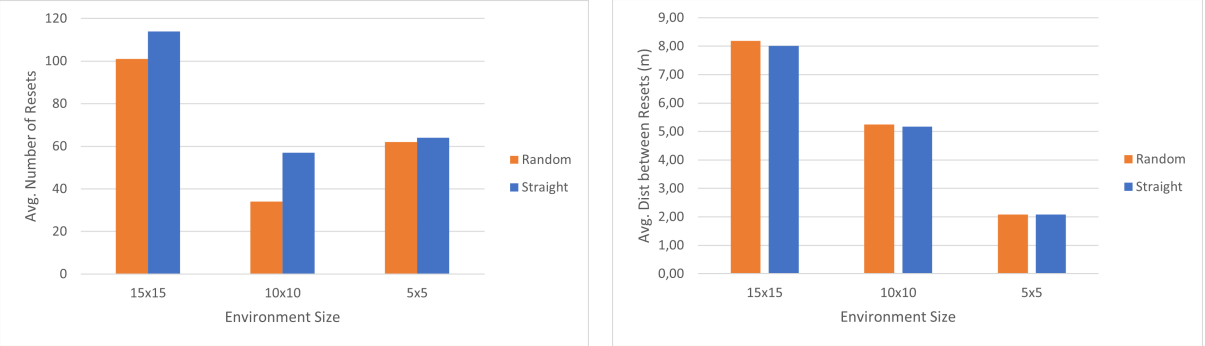
5. Results

5.1. How does APF perform?

The quantitative metrics, mentioned in Chapter 4, were used to analyse the performance of the APF algorithm in different scenarios.

5.1.1. Different Environments

Both the random and straight VEs were compared with regard to the average number of resets that occurred during the five minute runtime and the average distance a user travelled between resets, see Figure 5.1.



(a) Change in Average Number of Resets.

(b) Change in Average Distance to Reset.

Figure 5.1.: Comparison of environment types for virtual experiences with one user and without any improvements.

When looking at the number of resets, the random VE performed better for all sizes compared to the straight VE. As discussed in Chapter 4, the random VE consists of an approximately 200 m randomly generated path with slightly curved sections. Those curved sections cause the user experiencing an additional rotation to the recommended rotations of the APF algorithm. It results in a more curved physical path and consequently a higher distance between resets and lower number of resets compared to the straight VE.

For the 5×5m environment size, the difference is almost negligible (3.23 % for the number of resets and 0.17 % for the distance between resets). The reason for this can be seen in the physical paths plots of the user in Figure 5.2. The small environment does not give the APF algorithm enough space and time to properly veer the user in any of the recommended directions. Consequently, all the metrics for the two different VEs turned out to be roughly the same.

In Figure 5.2, it is seen that the user sometimes wanders outside the boundaries. This was caused by two things:

- Sometimes the user did not immediately react to a reset, resulting in him crossing the border.
- The APF algorithm uses a threshold to trigger the reset mechanism. This threshold sometimes was not properly adjusted to the small environments, resulting in the algorithm being too late to trigger the reset.

To solve the issue, the reset mechanism should be tweaked in a way that allows the user to still have some time to react before colliding with the wall.

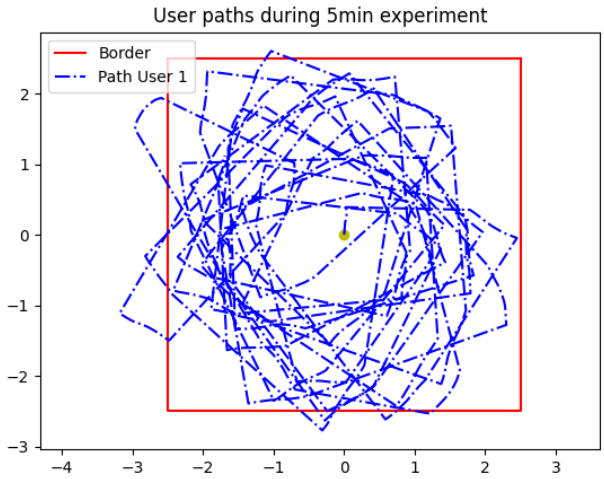


Figure 5.2.: Physical path of the user in a 5×5 environment.

The difference between the two environments can be observed in the 10×10 environment. This was certainly the case in terms of the number of resets, where we could see a decrease of 40.35 % for the random VE compared to the straight VE. As mentioned before, we assume the additional rotation provided by the curved path to be the cause for this improvement.

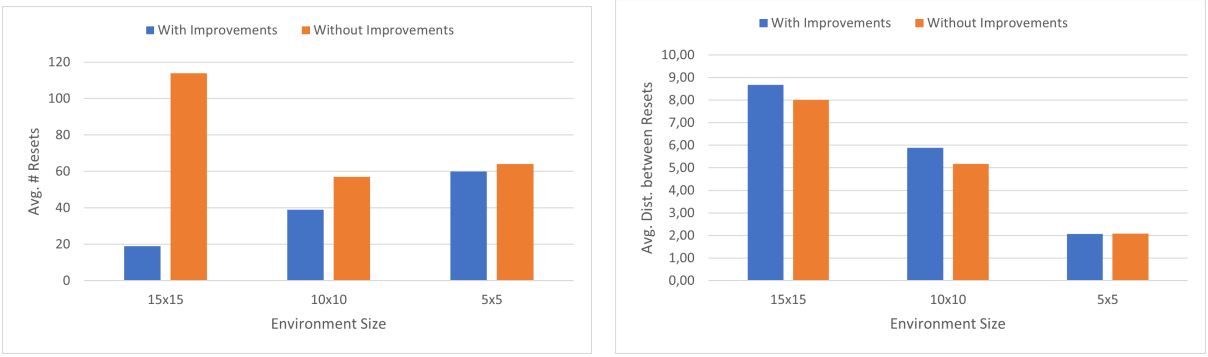
Surprisingly, our results showed an increase in the average number of resets at the 15×15

space compared to the rest. Intuitively the number of resets should be lower than the 10×10 environment, since a larger space accommodates more uninterrupted movement for the user. This was also confirmed by the RDW algorithm simulator built by Lemic et al. [57]. Additionally, the average distance between the resets showed that the movement area was used to its full potential by allowing a distance of more than 8 meters between resets. The reason behind this unexpected behaviour in the 15×15 space, lies in the problems APF has with the high frequency of positional updates (which we have addressed in the improvements). When a reset gets triggered, it takes a couple of seconds for the user to stop and rotate themselves while the HMD keeps sending positional updates to the central node. This resulted in the APF algorithm thinking the user is not reacting and, as a consequence, it keeping to recommend to the user to resets themselves. In reality the user visually only got one reset while the algorithm recommended multiple consecutive and identical ones.

We suspect that this behavior only happens in the 15×15 environment because of the *redirection radius* setting in the APF algorithm, that was set to 7.5 m. However, to be sure of this assumption, more tests in the 15×15 environment without improvements need to happen.

5.1.2. Improvements

Using the same parameters, we analysed the difference our improvements made on the performance of the RDW algorithms and APF specifically.



(a) Improvement in Number of Resets.

(b) Improvement in Average Distance to Reset.

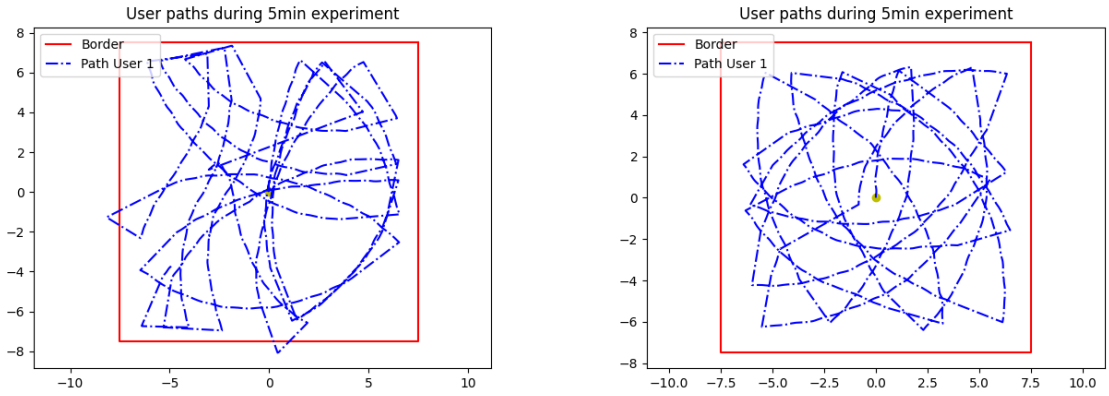
Figure 5.3.: Usefulness of suggested improvements for virtual experiences with one user in a straight environment.

Generally, from Figure 5.3, we can conclude that the improvements contribute to a better performance of the APF algorithm for both the straight and random VEs. Our improvements result in a reduction in the average number of resets of 83.33 % and a 31.58 % for the 15×15 and 10×10 environments, respectively. As for the average distance between resets, we managed to

increase the performance of the 15×15 environment by 8.36 % and the 10×10 environment by 13.93 %. Again, for the average number of resets without improvements, the discrepancy between the 15×15 and the 10×10 environment can be noticed. The reasoning behind this difference was explained in Section 5.1.1 and was solved with the improvements. However, we once more see that the APF algorithm in the 5×5 environment struggles to avoid a large number of resets, even with the improvements enabled.

Since the different improvement strategies were implemented simultaneously, we were not able to study the contribution of each improvement separately since this was feasible given the experiment runtime. However, two of our improvement strategies are expected to influence the metrics used in this section, namely the implementation of latency awareness and the reduction of border bouncing. All the other improvements, ignoring drift, head rotations and oscillations, focus more on the noticeability and overall experience of the user.

Figures 5.4a and 5.4b show the physical path one of the users (randomly selected) followed, using the algorithm without and with our improvements, respectively. At every reset point, we direct the user away from the walls and put APF on "cool down" for a very small amount of time. This is not the case in Figure 5.4a, where almost every reset triggers an immediate second reset consequently degrading the user experience and possibly introducing nausea [59].



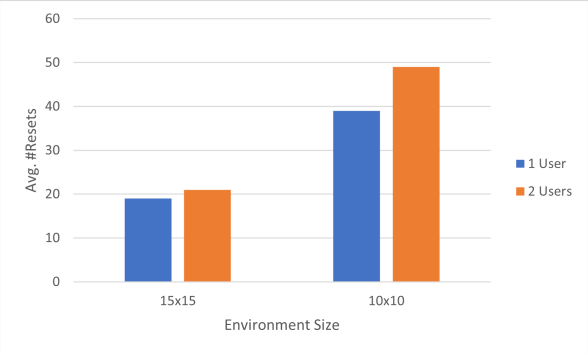
(a) Without improvements.

(b) With improvements.

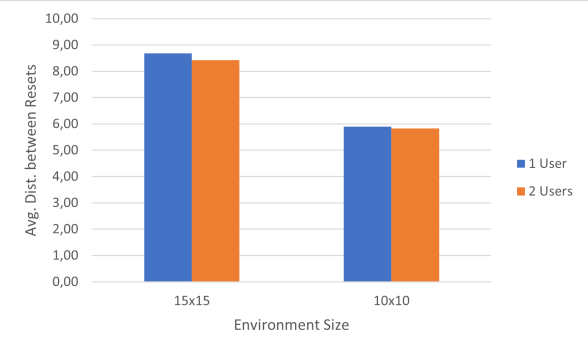
Figure 5.4.: Physical paths of the users in experiments running the straight VE for 5 minutes in a 15x15 environment.

5.1.3. Multiple Users

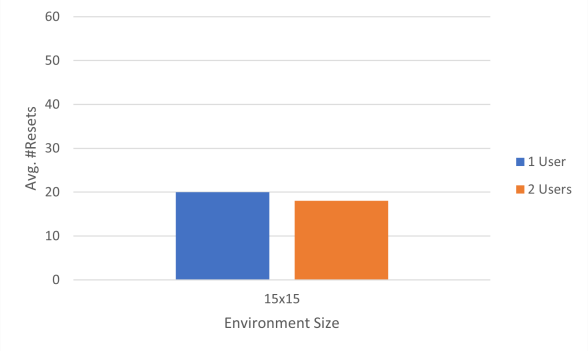
The impact of having more than one user was analysed. Because we could not comfortably accommodate more than one user in a 5×5 environment, this size was omitted from the experiment, as can be seen in Figure 5.5 and Table 4.2.



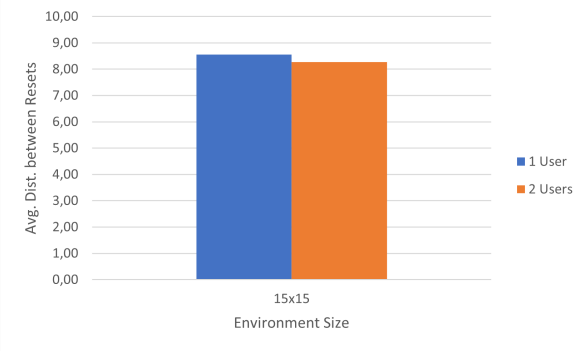
(a) Change in Average Number of Resets in a straight environment.



(b) Change in Average Distance to Reset in a straight environment.



(c) Change in Average Number of Resets in a random environment.



(d) Change in Average Distance to Reset in a random environment.

Figure 5.5.: Comparison of multiple users walking in a VE with improvements enabled while being in the same physical space.

For both metrics, our results were in line with previously described outcomes [57]: Apparently, having two users roaming in the same 15×15 physical space has little effect on the performance of APF. The algorithm managed to keep both users away from each other during the full duration of the experiment. APF however does not perform that well with more than one user in a 10×10 environment. According to the experimental data, one can expect an increase of 29.64 % in the number of resets for the 10×10 environment and 10.52 % in the 15×15 environment. The average distance between resets also changes: a decrease of 2.99 % can be expected in the 15×15 room compared to a 1.02 % decrease in the 10×10 environment.

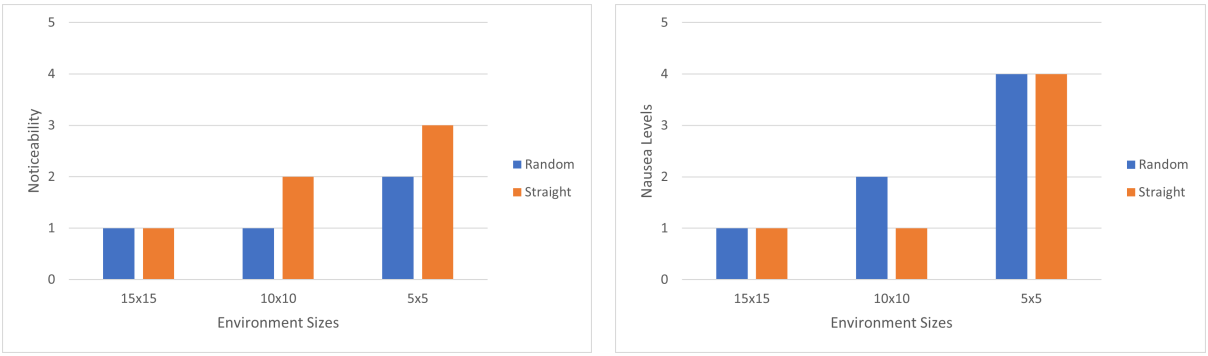
5.2. How noticeable is APF?

On top of the quantitative metrics, APF can also be analysed on user experience with qualitative metrics. In the following graphs, the median was always used instead of the mean, because we work with qualitative numeric data.

In all the different configurations and comparisons, it was clear that no scenario could unnoticeably redirect all the users during the five minute runtime (Likert score of 0). During the questioning of testers, it became clear that the reason behind this result lies with the background and VR experience 4 of our 7 subjects had. The APF algorithm was configured in a way that the noticeability threshold, determined by Steinicke et al., needed to be respected [45][43]. However, for test subjects with an IT development background or VR experience, this threshold was for some participants already noticeable. It compensated the measurements of the more naive users, who in certain configurations did not notice any redirection.

5.2.1. Different Environments

Firstly, different environments were compared with regard to noticeability and nausea levels, as shown in Figure 5.6.



(a) Change in Noticeability levels.

(b) Change in Nausea levels.

Figure 5.6.: Comparison of environment types for virtual experiences with one user and with the improvements activated.

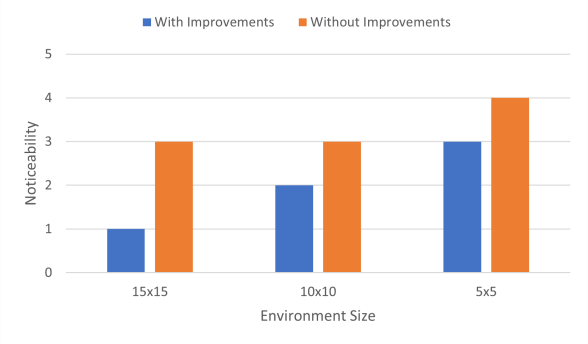
As was expected, the 15×15 environment performed well on both the noticeability and the nausea levels for the random and straight VEs. We mentioned before that the random environment adds a small amount of rotation to the already curved physical path. Our results indicate that this rotation was not large enough to reduce the noticeability of the user and only benefited the quantitative metrics from the previous section.

For the 10×10 environment, a difference was noticeable between the two VE types. The random rotation from the random VE did keep the noticeability as low as the 15×15 environment, while the straight VE struggled to unknowingly steer the user trough the smaller room. However, even though the extra rotation benefited the noticeability, the users had a tendency to get nauseous. This seemed to be caused by the combination of the APF algorithm having to more drastically steer the user and the extra rotation added by the curved path.

The quantitative metrics for the 5×5 environment indicated it was too small for one user to comfortably roam in while being steered. The results of both the noticeability and nausea levels confirmed this assumption. The high number of resets ($\pm$ 60 in a five minute runtime) meant the user had to drastically turn every five seconds, resulting in most of the test subjects getting nauseous. At this point, noticeability became irrelevant because users had to stop their virtual experience.

5.2.2. Improvements

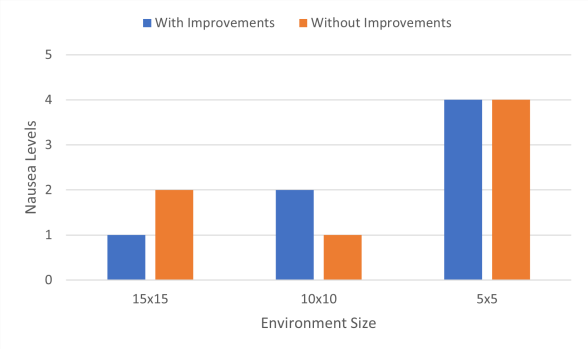
Secondly, the effectiveness of our improvements with regards to noticeability and nausea were studied. Figure 5.7, shows the results for the straight and random environments.



(a) Improvement in Noticeability levels in a straight environment.

(b) Improvement in Nausea levels in a straight environment.

(c) Improvement in Noticeability levels in a random environment.



(d) Improvement in Nausea levels in a random environment.

Figure 5.7.: Comparison of improvements for virtual experiences with one user.

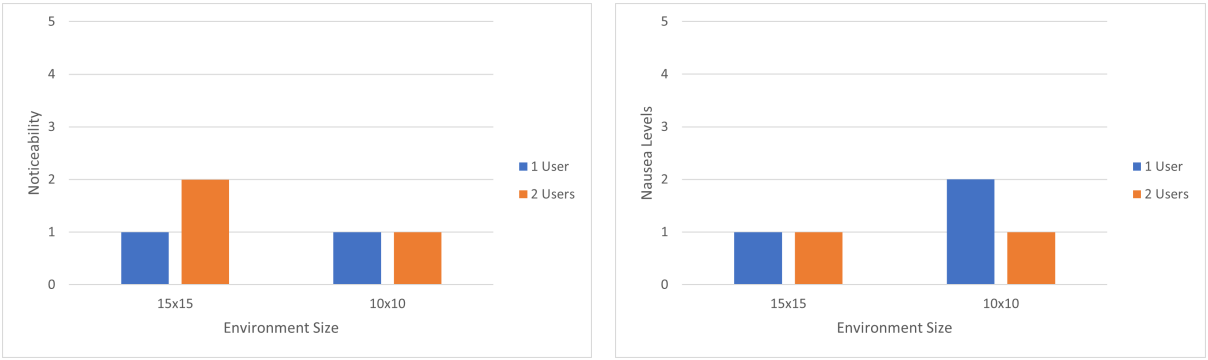
Overall, we can conclude that our improvements mostly targeted the noticeability rather than the nausea levels. The achieved reduction in noticeability varied from 1 point (5×5 and 10×10 environment) to 2 points (15×15 environment).

We mentioned before that we have implemented three improvements targeting the qualitative metrics: ignoring the drift, head rotations and oscillations. Again we cannot determine which of those improvements contributed the most. We assumed the oscillations, that would regularly appear during preliminary tests and tests without improvements activated, would contribute the most. One could argue that another of our implemented improvement strategies, namely the reduction of border bouncing, might have an impact on the noticeability levels for the 15×15 environment as well. This because the users are not being redirected for a small interval after a reset. However, the timeout interval was set very low and, when looking at the quantitative metrics in the previous section, did not seem to have that much of an impact on the possibilities for redirection. It mainly reduced the number of resets that occur, which we asked the subjects to ignore in their questionnaires.

In the 10×10 and 5×5 environment, we could identify similar trends as before. Again, for the 5×5 environment, the nausea levels were so high that the noticeability seemed to be irrelevant.

5.2.3. Multiple Users

Lastly, we investigated the change in noticeability when two users are roaming through the environment for five minutes at a time, as visually presented in Figure 5.8.



(a) Change in Noticeability levels.

(b) Change in Nausea levels.

Figure 5.8.: Comparison of multiple users in a random environment with improvements activated.

In Section 5.1.3, we concluded that the introduction of a second user in the environment did not lead to a difference in experience with regard to the quantitative metrics. However, when looking at the noticeability, the influence of having an extra user became apparent. This increase in noticeability might be due to the additional steering by the APF algorithm to push the users away from each other. This introduces more opportunities for the user to notice the redirection and break the immersion.

Surprisingly, we could not see this increase in noticeability with the 10×10 environment. However, given the limited time scope and low number of tests, it is possible that these results do not represent a real life scenario. Hence, it is possible that, in reality, no difference exists between the 15×15 and 10×10 environment, when it comes to the impact of a second user on the noticeability level. Moreover, the movement of users is always unpredictable, resulting in some tests having less user to user collisions and interactions compared to others.

6. Conclusion

Firstly, we developed a series of improvements, applicable to all RDW algorithms: removal of drift and head rotations, the reduction of border bouncing and oscillations and the introduction of latency awareness. The improvements were tested in various scenarios that varied in terms of the number of users, VE types and physical environment sizes. We observed that our improvements caused a decrease in the average number of resets and increased the average distance between resets for scenarios with both one and two users in the 15×15 and 10×10 environment. Additionally, we managed to decrease the noticeability levels in those same scenarios as well. It was clear that the 5×5 environment was too small for any type of unnoticeable and comfortable RDW experience. Furthermore, we concluded that the random VE performed better compared to the straight VE with regard to average number of resets, average distance between resets and noticeability. However, due to the extra rotation added by the environment, users would often experience an increase in nausea levels.

Secondly, a framework was constructed that enabled quick and easy testing of different RDW algorithms with multiple users and different VEs. Additionally, it collected data to determine the general experience of the user during the experiment. By performing experiments with multiple users in different VEs, we validated the usefulness of our framework.

Testing the noticeability and performance of an RDW algorithm with a standalone HMD in an open field, presented itself with some challenges. But, with the correct measures in place and the framework being adjusted to that environment, we managed to produce valuable data about the performance and noticeability of APF in different scenarios. The traces serve as an indication of the possibilities and limitations of APF.

Lastly, because we tested with only six people, we have to realize the randomness factor in these experiments. In the future, multiple repetitions of these tests would be needed to achieve statistical accuracy. Additionally, different RDW algorithms, VEs, communication protocols and positional extrapolators will need to be added to the framework and will need to be tested to provide us with a better overview of the optimal settings for RDW experiences.

This thesis was submitted for publication in the IEEE MultiMedia magazine.

A. References

[1] *Five years of VR: A look at the greatest moments from Oculus Rift to the Quest 2*. URL: https://www.oculus.com/blog/five-years-of-vr-a-look-at-the-greatest-moments-from-oculus-rift-to-quest-2/.

[2] *HTC to lead Virtual Reality with VIVE*. URL: https://www.vive.com/eu/newsroom/2016-06-09/.

[3] *PlayStation VR: Launch Lineup and Beyond*. URL: https://blog.playstation.com/2016/10/05/playstation-vr-launch-lineup-and-beyond/.

[4] *2016: the year when VR goes from virtual to reality*. URL: https://www.bbc.com/news/technology-35205783.

[5] *Virtual Reality Market Size, Share COVID-19 Impact Analysis*. URL: https://www.fortunebusinessinsights.com/industry-reports/virtual-reality-market-101378.

[6] Leila Meyer. "Students explore the earth and beyond with virtual field trips". In: *The Journal* 43.3 (2016), pp. 22–25.

[7] Tim Joda et al. "Augmented and virtual reality in dental medicine: A systematic review". In: *Computers in biology and medicine* 108 (2019), pp. 93–100.

[8] *About Us*. URL: https://vrallart.com/#about-us.

[9] Svetlana Bialkova and Marnix S Van Gisbergen. "When sound modulates vision: VR applications for art and entertainment". In: *2017 IEEE 3rd Workshop on Everyday Virtual Reality (WEVR)*. IEEE. 2017, pp. 1–6.

[10] Qinping Zhao. "A survey on virtual reality". In: *Science in China Series F: Information Sciences* 52.3 (2009), pp. 348–400.

[11] Dimiter Velev and Plamena Zlateva. "Virtual reality challenges in education and training". In: *International Journal of Learning and Teaching* 3.1 (2017), pp. 33–37.

[12] Eugenia M Kolasinski. *Simulator sickness in virtual environments*. Vol. 1027. US Army Research Institute for the Behavioral and Social Sciences, 1995.

[13] Joseph J LaViola Jr. "A discussion of cybersickness in virtual environments". In: *ACM Sigchi Bulletin* 32.1 (2000), pp. 47–56.

[14] *HTC Vive Pro Technical Specifications*. URL: https://www.vive.com/us/product/vive-pro2-full-kit/specs/.

[15] *PS VR Technical Specifications*. URL: https://www.playstation.com/nl-nl/ps-vr/tech-specs/.

[16] Tom De Schepper, Bart Braem, and Steven Latre. "A virtual reality-based multiplayer game using fine-grained localization". In: *2015 Global Information Infrastructure and Networking Symposium (GIIS)*. 2015, pp. 1–6. DOI: 10.1109/GIIS.2015.7347176.

[17] *Gear VR Technical Specifications*. URL: https://www.samsung.com/global/galaxy/gear-vr/.

[18] *Oculus Quest 2 Technical Specifications*. URL: https://store.facebook.com/be/quest/products/quest-2/tech-specs#tech-specs.

[19] *Delight VR - XR Glossary*. URL: https://delight-vr.com/xr-glossary/.

[20] *Omni One*. URL: https://omni.virtuix.com/.

[21] *System Requirements for the HTC Vive*. URL: https://www.vive.com/eu/support/vive/category_howto/what-are-the-system-requirements.html.

[22] Jan L Souman et al. "Walking straight into circles". In: *Current biology* 19.18 (2009), pp. 1538–1542.

[23] Sharif Razzaque. *Redirected walking*. The University of North Carolina at Chapel Hill, 2005.

[24] Sharif Razzaque et al. "Redirected walking in place". In: *EGVE*. Vol. 2. 2002, pp. 123–130.

[25] Niels Christian Nilsson et al. "15 Years of Research on Redirected Walking in Immersive Virtual Environments". In: *IEEE Computer Graphics and Applications* 38 (2018), pp. 44–56.

[26] *Connect 2021: Our vision for the metaverse*. URL: https://tech.fb.com/ar-vr/2021/10/connect-2021-our-vision-for-the-metaverse/.

[27] *OneBonsai - Cases*. URL: https://onebonsai.com/cases/.

[28] *Mazerspace - Products*. URL: https://mazerspace.com/.

[29] Jolanda Tromp et al. "Massively multi-user online social virtual reality systems: ethical issues and risks for long-term use". In: *Social networks science: design, implementation, security, and challenges*. Springer, 2018, pp. 131–149.

[30] Venkatakrishnan Parthasarathy et al. "Performance Evaluation of a Multi-User Virtual Reality Platform". In: *2020 International Wireless Communications and Mobile Computing (IWCMC)*. 2020, pp. 934–939. DOI: 10.1109/IWCMC48107.2020.9148390.

[31] Ryan R. Strauss et al. “A Steering Algorithm for Redirected Walking Using Reinforcement Learning”. In: *IEEE Transactions on Visualization and Computer Graphics* 26.5 (2020), pp. 1955–1963. DOI: 10.1109/TVCG.2020.2973060.

[32] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[33] Bruce Krogh. “A generalized potential field approach to obstacle avoidance control”. In: *Proc. SME Conf. on Robotics Research: The Next Five Years and Beyond, Bethlehem, PA, 1984*. 1984, pp. 11–22.

[34] Oussama Khatib. “Real-time obstacle avoidance for manipulators and mobile robots”. In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1985, pp. 500–505.

[35] Cole Hoffbauer. “Multi-user redirected walking and resetting utilizing artificial potential fields”. PhD thesis. Miami University, 2018.

[36] Eric R. Bachmann et al. “Multi-User Redirected Walking and Resetting Using Artificial Potential Fields”. In: *IEEE Transactions on Visualization and Computer Graphics* 25.5 (2019), pp. 2022–2031. DOI: 10.1109/TVCG.2019.2898764.

[37] M Zmuda et al. “Improved resetting in virtual environments”. In: *IEEE Virtual Reality*. 2013, pp. 91–92.

[38] Tabitha C Peck, Henry Fuchs, and Mary C Whitton. “Evaluation of reorientation techniques and distractors for walking in large virtual environments”. In: *IEEE transactions on visualization and computer graphics* 15.3 (2009), pp. 383–394.

[39] Betsy Williams et al. “Exploring large virtual environments with an HMD when physical space is limited”. In: *Proceedings of the 4th symposium on Applied perception in graphics and visualization*. 2007, pp. 41–48.

[40] Eric Hodgson, Eric Bachmann, and David Waller. “Steering immersed users of virtual environments: Assessing the potential for spatial interference”. In: *ACM: Transactions on Applied Perception* 8 (2011), pp. 1–22.

[41] Michael A. Zmuda et al. “Optimizing Constrained-Environment Redirected Walking Instructions Using Search Techniques”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.11 (2013), pp. 1872–1884. DOI: 10.1109/TVCG.2013.88.

[42] Tom Field and Peter Vamplew. “Generalised algorithms for redirected walking in virtual environments”. In: (2004).

[43] Frank Steinicke et al. “Estimation of Detection Thresholds for Redirected Walking Techniques”. In: *IEEE Transactions on Visualization and Computer Graphics* 16.1 (2010), pp. 17–27. DOI: 10.1109/TVCG.2009.62.

[44] Frank Steinicke et al. “Analyses of human sensitivity to redirected walking”. In: *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*. 2008, pp. 149–156.

[45] Frank Steinicke et al. “Moving towards generally applicable redirected walking”. In: *Proceedings of the Virtual Reality International Conference (VRIC)*. 2008, pp. 15–24.

[46] Eric Hodgson and Eric Bachmann. “Comparing Four Approaches to Generalized Redirected Walking: Simulation and Live User Data”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.4 (2013), pp. 634–643. DOI: 10.1109/TVCG.2013.28.

[47] David Waller et al. “The HIVE: A huge immersive virtual environment for research in spatial cognition”. In: *Behavior research methods* 39.4 (2007), pp. 835–843.

[48] Filip Lemic, Jakob Struye, and Jeroen Famaey. “Short-Term Trajectory Prediction for Full-Immersive Multiuser Virtual Reality with Redirected Walking”. In: (2022).

[49] Georgios Tsaramirsis et al. “Navigating virtual environments using leg poses and smartphone sensors”. In: *Sensors* 19.2 (2019), p. 299.

[50] Matthew O’Riordan. “Everything You Need To Know About Publish/Subscribe”. In: (). URL: https://ably.com/topic/pub-sub.

[51] *Paho Overview*. URL: https://wiki.eclipse.org/Paho.

[52] *What can rabbitmq do for you?* URL: https://www.rabbitmq.com/features.html.

[53] *Why ZeroMQ?* URL: https://zeromq.org/.

[54] *VrApi Fundamentals*. URL: https://developer.oculus.com/documentation/native/android/mobile-vrapi/.

[55] Zhuangwei Kang et al. “Evaluating DDS, MQTT, and ZeroMQ Under Different IoT Traffic Conditions”. In: (2020).

[56] Guo Fu, Yanfeng Zhang, and Ge Yu. “A Fair Comparison of Message Queuing Systems”. In: *IEEE Access* 9 (2020), pp. 421–432.

[57] Filip Lemic, Jakob Struye, and Jeroen Famaey. “User Mobility Simulator for Full-Immersive Multiuser Virtual Reality with Redirected Walking”. In: *Proceedings of the 12th ACM Multimedia Systems Conference*. MMSys ’21. Istanbul, Turkey: Association for Computing Machinery, 2021, pp. 293–299. ISBN: 9781450384346. DOI: 10.1145/3458305.3478451. URL: https://doi.org/10.1145/3458305.3478451.

[58] Eric Hodgson, Eric Bachmann, and Tyler Thrash. “Performance of Redirected Walking Algorithms in a Constrained Virtual World”. In: *IEEE Transactions on Visualization and Computer Graphics* 20.4 (2014), pp. 579–587. DOI: 10.1109/TVCG.2014.34.

[59] Jason Jerald, Mary Whitton, and Frederick P Brooks Jr. “Scene-motion thresholds during head yaw for immersive virtual environments”. In: *ACM Transactions on Applied Perception (TAP)* 9.1 (2012), pp. 1–23.

[60] Claudiu-Bogdan Ciumedean et al. “Mission impossible spaces: Using challenge-based distractors to reduce noticeability of self-overlapping virtual architecture”. In: *Symposium on Spatial User Interaction*. 2020, pp. 1–4.

[61] Ernst Kruijff et al. “The influence of label design on search performance and noticeability in wide field of view augmented reality displays”. In: *IEEE transactions on visualization and computer graphics* 25.9 (2018), pp. 2821–2837.

[62] Jingxi Xu and Benjamin W Wah. “Consistent synchronization of action order with least noticeable delays in fast-paced multiplayer online games”. In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 13.1 (2016), pp. 1–25.

[63] Sarthak Ghosh et al. “NotifiVR: exploring interruptions and notifications in virtual reality”. In: *IEEE transactions on visualization and computer graphics* 24.4 (2018), pp. 1447–1456.

[64] Joel Weidenmark. *Acceptable Ads guidelines, its effect on user experience and ad-noticeability*. 2020.

[65] David Geerts et al. “Are we in sync? synchronization requirements for watching online video together.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2011, pp. 311–314.

[66] S. A. McLeod. *Likert scale*. URL: `www.simplypsychology.org/likert-scale.html`.

[67] Ankur Joshi et al. “Likert scale: Explored and explained”. In: *British journal of applied science & technology* 7.4 (2015), p. 396.

[68] Gustav Theodor Fechner. *Elemente der psychophysik*. Vol. 2. Breitkopf u. Härtel, 1860.

[69] Kjetil Raaen and Ivar Kjellmo. “Measuring Latency in Virtual Reality Systems”. In: *Entertainment Computing - ICEC 2015*. Ed. by Konstantinos Chorianopoulos et al. Cham: Springer International Publishing, 2015, pp. 457–462. ISBN: 978-3-319-24589-8.