Hindawi
Advances in Human-Computer Interaction
Volume 2023, Article ID 1563506, 18 pages
https://doi.org/10.1155/2023/1563506

Research Article

Hand-Controller Latency and Aiming Accuracy in 6-DOF VR

Viktor Kelkkanen, David Lindero, Markus Fiedler, and Hans-Jürgen Zepernick

Blekinge Institute of Technology, Karlskrona, Sweden

Correspondence should be addressed to Viktor Kelkkanen; viktor.kelkkanen@bth.se

Received 23 March 2023; Revised 31 July 2023; Accepted 23 August 2023; Published 25 September 2023

Academic Editor: Marco Porta

Copyright © 2023 Viktor Kelkkanen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

All virtual reality (VR) systems have some inherent hand-controller latency even when operated locally. In remotely rendered VR, additional latency may be added due to the remote transmission of data, commonly conducted through shared low-capacity channels. Increased latency will negatively affect the performance of the human VR operator, but the level of detriment depends on the given task. This work quantifies the relations between aiming accuracy and hand-controller latency, virtual target speed, and the predictability of the target motion. The tested context involves a target that changes direction multiple times while moving in straight lines. The main conclusions are, given the tested context, first, that the predictability of target motion becomes significantly more important as latency and target speed increase. A significant difference in accuracy is generally observed at latencies beyond approximately 130 ms and at target speeds beyond approximately 3.5°/s. Second, latency starts to significantly impact accuracy at roughly 90 ms and approximately 3.5°/s if the target motion cannot be predicted. If it can, the numbers are approximately 130 ms and 12.7°/s. Finally, reaction times are on average 190–200 ms when the target motion changes to a new and unpredictable direction.

1. Introduction

Remotely rendered virtual reality (VR) enables high-fidelity graphics on thin devices that are otherwise limited by their computational capability due to physical constraints. One problem with remote rendering for VR is the low latency that is required between input and output, or motion-to-photon (MTP) latency, a requirement of which is generally stated to range between 7 and 20 ms [1–7]. However, by utilizing latency-mitigation techniques such as prediction [8, 9] and just-in-time image warping [9–15], a significant amount of latency can still be tolerable in remote VR depending on content, even up to 90 ms according to one study [16]. So far, prediction and warping have been widely deployed in terms of headset orientation and translation, making head motions seem near-instant even during significant network delays. For hand controllers, prediction is typically utilized while warping is more difficult since the controllers are individual 3D objects that may be occluded by or occlude other 3D objects in the rendered image. The state-of-the-art in 2022 3D image warping that considers individual 3D objects in the scene were so far able to compute new frames in the time-scale of seconds [17, 18]. While performance may be improved in the future, we have not yet seen an implementation that would be viable for computing VR hand-controller warping in real-time. Although prediction will mitigate some latency, it cannot provide the perfect responsiveness that image warping enables. This is because the accuracy of prediction will deteriorate with increased extrapolation ranges (latency) and sudden, unpredictable motions (of present-day sensors) cannot be predicted at all. Late warping on the other hand has access to the latest input data and can therefore adjust accordingly with a short extrapolation range when used just before scan-out to the display.

The negative effects of latency on the performance and presence [19] in remote manipulation have been studied since the 1950s [20, 21]. The high latencies associated with space exploration tasks generated a strong interest in this field during the late 1950s and throughout the 1960s by NASA and other parties [22–24]. A common topic of this era was the problem of controlling remote vehicles, e.g., while

operating on the moon [22]. In the 1970s, research on the topic expanded to include the context of aircraft piloting, commonly with the purpose of developing realistic flight simulators using computer-generated images with low latencies between controller and visual output [21, 25, 26]. Throughout the 1980s and 1990s, research expanded further to include head mounted displays (HMDs) for pilots [8, 10, 21, 27]. Critical knowledge was discovered and refined during this era that enabled the modern VR headsets of today, for example, works on prediction for HMDs [8], image deflection (warping) [10], and methods for tracking objects in a physical space [28]. In the present era, commercial VR headsets enable near-zero head-tracked latencies by utilizing modern hardware and advanced latency-mitigation techniques. Nonetheless, the delayed actuation of controller inputs still cannot be mitigated in a general way. The emerging use case of remote VR through potentially high-latency networks makes the problem relevant also in this present era of VR.

To solve the issue of subpar hand-controller responsiveness, split rendering has been proposed for remote rendering applications [29–31]. In such solutions, the background and otherwise noninteractive objects are rendered on the server, while interactive objects are rendered on the client. The two are finally composited before presenting the frame on the client. While minimizing hand-controller latency, the split-rendering methods require the development of special clients for each application which defeats the purpose of remote rendering to some degree. Because interactive objects are rendered from geometry on the client, their visual fidelity is limited to the rendering capabilities of the client hardware. Ideally, this should be avoided since the purpose of remote VR is typically to provide high-fidelity graphics on otherwise limited devices. There are other use cases where increased visual fidelity is not the purpose, e.g., in remote operation of physical actuators where the VR view is a live video feed. However, in such a scenario, it seems unlikely that any visual modification of the actuator is desirable.

While hand-controller latency can be mitigated visually by, for example, present split-rendering or potential future 3D-image warping methods, the problem of delayed remote control remains. Typically, there is some game engine or physical actuator that is controlled remotely by the delayed input signals. That delay depends mainly on the network and codec [32], and it may be subject to physical constraints that cannot be solved without moving the actuation to the client. Moving the actuator may not be possible for physical applications but it could be done for some types of software by moving the affected logic to the client. For example, if the client is playing a game remotely, the aiming logic may be moved to the client if it can be trusted. Regardless, we may assume that visual issues due to latency can be solved by future engineering endeavours, but actuator latency will still remain due to physical constraints and applications where logic must be computed on the server. Thus, the following question emerges: Given a visual latency otherwise equal to local VR, which tasks can be operated remotely and at what hand-controller latency? If the task requires a motion of X°/s with a maximum error in accuracy of Y°, what is then the maximum additional network-induced latency that can be allowed on the remote VR system? We hypothesize that some tasks are less sensitive to latency, in particular when the required hand-controller motion can be predicted by the user, and most importantly, if slow motions are sufficient to accomplish the task. On the other hand, tasks that require significant speed and when motion cannot be predicted are likely more difficult to accomplish at high latencies. In summary, the purpose of this work is to answer the following research questions (RQs):

(1) RQ1: How do the target speed and hand-controller latency affect the aiming accuracy in VR?

(2) RQ2: How is aiming accuracy affected if the target direction can be predicted by the VR operator?

The main contributions are rooted in the collected and presented data. We expand upon the current literature by conducting an experiment untied to a specific task other than aiming at a target that changes direction and moves in straight lines in VR. Furthermore, we provide more fine-grained objective results on accuracy in relation to latency as compared to subjective evaluation and specific task objectives.

The remainder of the article is structured as follows: Related work is discussed in Section 2. A basic test that confirms the hand-controller latency reported in literature regarding the HTC Vive VR system is included in Section 3. Section 4 describes the main experiment of the study. Section 5 presents the results of the experiment and provides an accompanying discussion tied to each part of the results. The results are concluded in Section 6. Finally, limitations and future work are discussed in Section 7.

2. Related Work

Hand-controller latency and accuracy in VR have been studied in previous work, but seldom in combination. When they were combined, the results found so far are generally tied to some specific task which makes them difficult to generalize to other applications. Our contribution is a result that may be generalized based on target motion speed and size at given latencies in contexts where the target motion can be predicted and when it cannot. The generalization is applicable to scenarios where the target keeps changing direction and moves in straight lines.

2.1. Aiming for Latency in VR: User Experiments. In 2011, a related work was published that studied virtual hand interactions in a VR environment with injected input latency [33]. In the study, an experiment was conducted in which users were tasked with moving their virtual finger from one button to another. The finishing button varied in size and the authors found that users needed more time to press smaller buttons as latency increased. The authors concluded that it is therefore important to avoid using small targets when input latency is present in VR. Another finding from the study was that task completion time increased with latency, starting

from the lowest tested latency at an additional 55 ms on top of a system-inherent latency of 63 ms.

A related work on VR hand-controller latency published in 2019 studied the subjective quality of experience (QoE) of operating a forestry crane with added joystick latency [34] of 0, 50, 100, 200, 400, and 800 ms. In summary, no significant difference in subjective quality could be shown at latencies of up to 200 ms. In most cases, but depending on the specific quality parameter in question and user experience level, 800 ms were required to cause a significantly lower mean opinion score (MOS) as compared to cases without additional latency. Regarding the task, crane motion speed is not mentioned in the paper, but an average time consumption of approximately 13 s per lifted log can be derived from the reported data. We may expect that the logs are relatively large targets, thus decreasing the sensitivity to latency, but the exact target size is not reported in the paper.

A related work from 2021 studied the effects of controller latency in specific flight simulator combat tasks using the HTC Vive Pro [35]. The author tested latencies of 250, 500, 750, 1000, and 1250 ms with pilots and found a significant effect on "combat score" at latencies of 500 ms and beyond. Thus, also in this case, relatively high latencies were acceptable. However, several factors related to the task may have had a significant impact on this result. For instance, the pilots faced AI-controlled adversaries operating inferior aircraft. A future research question is whether latency would have a more significant effect if equally skilled pilots faced off in equal aircraft but with different latencies.

A related work from 2019 studied the objective user targeting accuracy when using 120 Hz eye-tracking and an HTC Vive hand-controller [36]. In summary, the study found that (1) Random movement of the target has a significant negative impact on accuracy when using a controller. (2) Target speeds of 12, 24, and 36°/s corresponded to average hand-controller accuracies of roughly 0.8–1.1°, 1.1–1.5°, and 1.5–1.8°, respectively, depending on the target path (linear, parabola, or random). For static, nonmoving targets, the accuracy was roughly 0.6°. The size of the target was 1.57° vertically and horizontally. It should be noted that only the random path condition contained turns in other directions in this related work. The random turns were furthermore smooth on a spline-form, contrary to the sharp edges used in this work, which may impact the overall accuracy.

Reporting the accuracy in degrees provides a generalized result that is applicable to other displays, and we use that format also in this work. For reference, the HTC Vive has a pixel density of 10.68×9.68 [37] pixels per degree (PPD). According to these data, the target size of 1.57° in the related work [36] was approximately 16×15 pixels.

2.2. Aiming with Latency: Other Devices. While user studies on hand-controller latency with modern VR equipment are relatively rare, plenty of research has been conducted regarding the vast field of input latency in human computer interfaces (HCI) that dates back to at least 1968 [38]. We present some recent and interesting results in the context of this study from related work below. A more comprehensive literature review of the field from 2017 is available in [39].

2.2.1. Haptics. A study published in 2007 showed the effects of latency on a collaborative task when using a haptic device [40]. Two users controlled one circle each in a networked program and were asked to "meet," stick together, and move in union towards a common goal position. Performance started decreasing from the lowest added latency of 25 ms and beyond, but users generally did not perceive the degradation until at 50 ms. Performance decreased steadily until around 100 ms when users began to move more slowly to compensate for the latency, thus preventing the error from increasing further. While accuracy remained largely the same beyond 100 ms, users still reported an increasingly difficult and more disruptive experience in proportion to the added latency up to the tested 400 ms. Indeed, these results highlight the human ability to adapt to latency by reducing movement speed while still finding the experience increasingly more annoying.

Another study on delayed haptic input published in 2005 found that haptic delay had little effect on performance as compared to visual delay in a task that involved tapping two targets as quickly as possible [41]. Results indicated that visual latency had an increasingly negative effect on all three performance parameters (mean intertap interval, mean number of target misses, and mean difficulty rating) starting from the lowest tested added latency of 25 ms. Haptic latency on the other hand did only slightly affected one parameter, the mean intertap interval, starting at the maximum tested latency of 150 ms.

2.2.2. Perceptible Latency. A study from 2014 estimated the perceptible latency while drawing with a pen and found that a latency of around 50 ms is perceptible on average when the hand and pen can be observed visually [42]. However, if the hand and pen were hidden from sight, the average perceptible latency doubled to around 100 ms. Thus, it seems that the presence of a visual reference may make latency about twice as noticeable.

A study on the perceptible input latency of touch screens was published in 2012 which showed that users are able to perceive latencies of 1 ms (and perhaps lower) on touch hardware [43]. Qualitative feedback from the study indicated that users commonly found that the experiment "broke" them, because they could no longer find the latencies of commodity devices acceptable after having done the experiment. These qualitative data hint at the adaptability of the human visual system in that we may learn to live with latency and not notice it until a reference is available. Anecdotally, this effect can be observed when using monitors with high refresh rates beyond 60 fps and subsequently viewing content at 60 fps or lower. A stutter may then become ever more apparent in any moving content.

2.2.3. Remote Surgery. While typically viewed on a regular monitor and not conducted in VR, remote surgery is a related field where robotic surgical tools are operated remotely

through a network. The feasibility of the technology was shown in 2002 when a successful surgery was achieved at a distance of 14000 km with a mean delay of 155 ms [44]. The authors stated that a latency below 300 ms should be feasible for remote surgery, according to their research [44].

In 2005, 21 remote surgeries were reported to have been successfully conducted at a distance of 400 km with a latency of 135–140 ms [45]. Out of the 140 ms latency, only 14 ms were due to network transmission; the rest were added by the video codec. The authors note that the latency was perceivable by the surgeons, but that they were able to adapt their motions to compensate for the delay.

The consensus varies regarding the latency requirements of remote surgery. The authors in [44] found 300 ms to be the upper limit. But a study published in 2008 found that a delay of 450 ms was considered manageable and that a 900 ms latency was cumbersome but could be overcome with deliberation by the surgeon [46]. In 2007, a study found that the acceptable latency largely depends on the task and that a large variation can be observed when asking surgeons whether surgery would be possible given the specific task and latency [47]. When mixing all tasks, 21% of surgeons stated that it would not be possible to do the surgery at the lowest studied latency of 150 ms. At 350 ms, that percentage increased to 62%. A study published in 2016 found further evidence of the impact of the task, and results suggested a reduced surgical effectiveness at total latencies beyond 160 ms [48].

Based on the related work, it seems that the feasibility of remote surgery depends not only on the latency but also on the surgeon and the given task. Furthermore, when studying the available literature in the field, it is not always clear which latencies are reported in the publications. Reporting varies from showing only the network transmission latency to showing the entire loop from motion to visual feedback at the surgeon's end [49]. A literature review from 2022 suggests that future efforts should improve reporting of the signal latency and follow careful research methodology [49].

A future research question is how remote surgical tasks can be generalized in terms of difficulty in connection to latency. What is it that makes a surgical task more or less sensitive to latency? Herein, we hypothesize that target speed and the ability to predict the upcoming motion are the most significant factors that can be used to generalize the difficulty of this category of aiming with latency tasks.

In this work, tests are conducted regarding the accuracy of users aiming with delayed hand-controller poses on moving targets in VR. However, we estimate that the experiment in its current state is too coarse to evaluate surgical tasks and that the tracking system of the consumer hardware is anyway too crude for surgery. Nonetheless, the overall method may be applicable to surgery in future work given a more granular experiment environment and more accurate tracking equipment.

2.3. VR: System Latency Estimations. Experiments in this work were conducted with the HTC Vive. In order to provide a generalized result, it is important that the inherent latency of this system is known. In this section, related work is presented that studied the latency of the HTC Vive. In Section 3, basic measurements were also conducted in-house to confirm that the numbers presented in the related work are within reasonable error limits.

2.3.1. HTC Vive Update Rates. The HTC Vive uses the Lighthouse tracking system for tracking its controllers and headset. This tracking system is based on the Minnesota Scanner [28], developed by Sorenson et al. in 1989 [50]. In short, a plastic box with one transparent side houses an infrared (IR) LED array, and two motors are used to emit sweeping light beams along the X and Y axes. The light is picked up by sensors built into the VR equipment which are able to position themselves in space based on the timings of the sensor activations. According to measurements by a third party, the Lighthouse motors rotate at 3600 rpm each (equivalent to 60 updates per second) [51]. The flashes are interleaved, resulting in horizontal and vertical updates separated by 8.33 ms (16.66 ms for one complete update). Between the flashes of 120 Hz, the equipment additionally utilizes dead-reckoning, or extrapolation, based on built-in inertial measurement units (IMU) [51]. The driver-internal update rate has been measured at approximately 1000 Hz (1 ms) for the headset and 360 Hz (2.77 ms) for the hand controllers in the internal OpenVR interface [51]. For the client OpenVR API, however, the update rate was measured at 225–250 Hz (~4 ms) [51].

2.3.2. Academic Research. A publication from 2022 presented the inherent hand-controller latencies measured in common consumer VR systems [52]. In summary, hand-controller latencies were measured for the HTC Vive, Oculus Rift, Rift S, and Valve Index VR systems. The work shows the different capabilities of the built-in motion-estimation components of each system and how the latency changes when the systems are able to estimate the upcoming motion. For example, the HTC Vive controller is shown to have an average MTP latency of around 31 ms for sudden movements and around 3.6 ms for continuous motions. The best system is shown to be the Oculus Rift, with an average sudden-movement latency of 20.6 ms and a continuous-movement latency of 1.5 ms.

In contrast to hand-controller latency, the latency related to the VR headset motion has been more widely studied. According to a paper from 2017, the upper-bound latency of the HTC Vive headset is 22 ms [53]. Thus, the HTC Vive headset itself appears to have faster tracking than its hand controllers. A similar result of 21.7 ms on average was shown in another study from 2019 [54].

Although the HTC Vive MTP latency was determined to be 22 ms for the headset and at least 31 ms for the controllers with sudden motions, additional latency may still be introduced by the application running the VR program. In a study on the HTC Vive Pro from 2018 [55], the authors found that the unity game engine introduced around one extra frame time of latency for a total of ~ 31.33 ms when compared to directly accessing the OpenVR API through

a simple Python program (~18.35 ms). That latency was defined as "app-to-photon" and is based on sending a signal from the game loop of either program and waiting for it to be indicated visually on the display by flashing the screen from black to white.

Another study of the HTC Vive tracking system measured the MTP latency of an individual HTC Vive tracker component, which is a general-purpose device that can be used to track a physical object in VR. The MTP latency of such a device was measured at 56.14 ms on average, when using Unreal Engine 4 as basis for the VR application [56].

3. Basic Latency Estimate of the HTC Vive

To confirm the numbers regarding the HTC Vive latency given in the related work, some basic measurements were conducted in-house and are presented in this section.

3.1. Pose Update Rate. The OpenVR function, GetControllerStateWithPose(), used in [51] was tested from the public client API [57]. Although deprecated by the time of writing, the documentation describes the function as polling the most recently updated controller pose. Indeed, the average update interval is measured at 3.9955 ms (250 Hz) based on approximately 25000 samples. However, an unnatural characteristic of spikes can be observed at the 2, 4, and 6 ms intervals in the distribution of the time consumption between successful polls where positions were not identical (see Figure 1). Based on the measured distribution, it seems that the system polls poses with a 2 ms interval (500 Hz), but that they will be updated only every 4 ms (250 Hz). This result was obtained on Windows 10 running Steam VR 1.23.7 on both HTC Vive (Base Station 1.0) and the valve index headset (Base Station 2.0).

The measurements support the minimal delay found for the HTC Vive in [52] (3.6 ms on average for continuous motion with a standard deviation of 3.9 ms). Indeed, it seems reasonable to assume that the best possible hand-controller latency for the HTC Vive is 4 ± 2 ms given the measurement results and related work.

3.2. Controller MTP. Moving on to measurements of the MTP for sudden hand-controller motions, a simple experiment was conducted where the hand-controller and headset display were both visible in a camera recording at 480 fps (see Figure 2). To capture the MTP, the headset display was rendered either all green (if the controller moved more than 1 mm since the previous frame) or all red (otherwise) depending on the present hand-controller motion. Thus, we deduce the MTP latency by pushing the controller and counting the number of frames from the start of its motion until a green image is shown on the display. Although usually more refined, this overall method of measuring MTP is common and used for example in [52].

With this rough estimation of the MTP latency, results were obtained in the range 33.3–47.9 ms (39.2 ms on average) for the valve index and 27.1–47.9 ms (35.7 ms on average) for the HTC Vive with an error of at least 2 ms due to using a 480 fps camera. The numbers do not match exactly those from the related work [52] where the index at 90 fps was measured at 37.5 ms average (~2 ms difference) and the HTC Vive at 30.8 ms average (~5 ms difference). However, based on the related work and additional estimations, it is safe to assume that sudden motion of the controller will generally produce a visual result 3-4 frames later at 90 fps on the HTC Vive. Therefore, to determine the total MTP latency, four frames of latency should be added to any artificially injected latency in the experiments conducted in this work. Note also that for precise calculations, 11.169 ms should be used as the frame-interval time as the exact frame rate is actually 89.53 fps in the HTC Vive and not 90 fps [58] as used in writing for the sake of brevity.

4. Methods

To answer the research questions, an experiment was designed in which human participants played a simple VR game containing the task of aiming with a virtual laser pointer that was projected with a varying degree of input latency in the 3D VR world. To be clear, latency is only injected into the laser dot and target position; headset and controllers are rendered no different than from standard local VR. Thus, a perfect visual warping is assumed but with a remaining delay on actuation. Visual examples of the laser dot and target are available in Figure 3. The game program is based on Valve's example code for Open VR [57] and was developed in C++ with OpenGL.

4.1. Experiment. During the experiment, a target moves in a random pattern along a wall at a speed randomly selected from a set of test conditions. The random pattern is constrained to consist of five lines whose length scales linearly with the speed. The users are tasked with aiming the hand controller as accurately as they can towards the center of the target while it moves along the wall. The hand controller acts as a laser pointer that renders a red dot at the location where the laser beam hits the wall or target. The wall and target are located in front of the user in the 3D world at Z coordinate −7.5, while the user is located at 0 (with some deviation due to physical movement and arm length). This represents a distance of approximately 7.5 m between the eye and the target.

When the test is running, a score is accumulated for each frame based on how close the laser dot is to the center of the target. The goal of the user is to achieve a score as high as possible by keeping the laser on-point while the target is moving. The immediate accuracy is continuously presented in discrete steps to the user in the range 0–11 (see Figure 3) for examples. Meanwhile, data such as angular accuracy are stored in the background for later analysis.

In addition to varying latency and target speed, one additional parameter was added that toggles the rendering of lines that show the upcoming path of the target (see Figure 4). This parameter adds coverage for use cases in which the user may or may not be able to predict in what direction the target will move after the next junction. For an

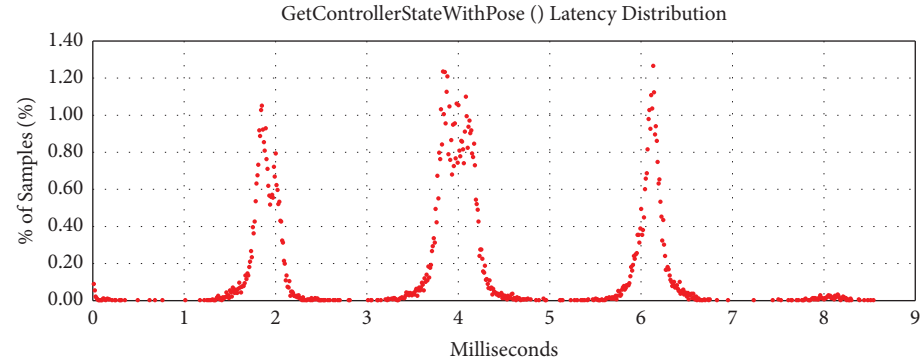


Figure 1: Distribution of the time consumed between polls to GetControllerStateWithPose() where the polled position was different from the previous one.

Figure 2: Frame of physical impact (left) with corresponding green frame indicating movement (right) displayed 15 camera frames later. The resulting MTP delay is thus approximately 31 ms due to recording at 480 fps. This test confirms that our hardware behaves roughly the same as reported in the related work.

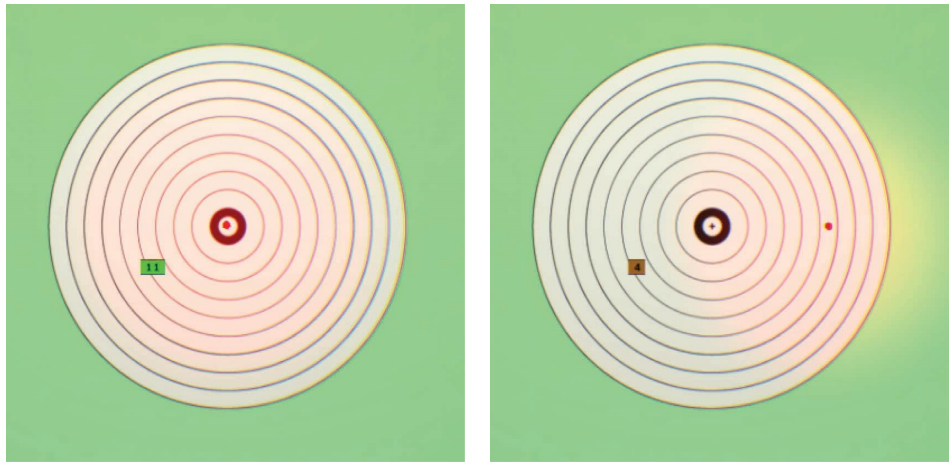
overview of all parameters included in the test (see Table 1). Upper and lower limits for the parameters speed and added latency were chosen based on previous experiments [59] and a supplementary testing made while developing the system. A logarithmic division of the parameter space increases precision in the region closer to 0 and makes larger jumps between higher values where the exact number is not as important anymore.

In summary, the experiment tests every speed with every latency, both with and without rendering the path, for a total of 96 different parameter combinations (test condition triplets). A single run tests one condition triplet. For example, the hand-controller latency may be set to 8 frames, and the target moves with a speed of 1 m/s while the upcoming path is rendered. During the run, the direction of the target will change five times and run for an average of 520 frames regardless of speed for a total of almost 6 s per run at 90 fps. See Figure 4 for an example of a run and its five random junctions. The total length of runs is kept nearly identical by adjusting the length of the lines linearly based on the speed that will be used during that run. To clarify, low speeds yield short lines but running those lines takes the same amount of time as running at higher speeds with longer lines. The number of frames will vary only slightly due to low-level errors such as floating point precision limits and potentially due to random frame misses that may occur in a Windows 10 consumer-grade system.

In the end, the user will spend around 10 minutes in the main part of the VR experiment ($6\,\text{s} \times 96 / (60\,\text{s/min}) = 9.6\,\text{min}$). However, the exact time depends on how long the user waits before starting each run. In addition, there is a training session containing eight conditions and a half-time pause of five minutes. A pause is a common practice in user experiments and was included to reduce the risk of exhausting users and to help them maintain focus during the entire experiment. To start the next run, the user must aim at the target center and press the trigger of the controller. The requirement of centering before each start ensures that the user is ready for the next run and that all tests start in similar conditions. For a visual overview of the experiment program see Figure 5.

Objective data are continuously gathered each frame the target is active for each run of each user session. In the end, conclusions are based on the statistics of the collected data, and the effect on accuracy of the tested parameters is determined.

4.2. Participants. The experiments were conducted at the Blekinge Institute of Technology at campuses Karlskrona (7 users) and Karlshamn (8 users) as well as at Ericsson Research in Luleå (10 users). A total of 25 users participated in the study, aged 19–63. Details of the demographic data are presented in Figure 6. All users signed an informed consent form that provided the details of the experiment. They were further informed that they may abort the experiment at any time for any reason and should not participate if they are prone to experiencing simulator sickness. The shared VR equipment was sanitized with disinfectant between each user and disposable face pads were offered for use with the VR headset. Finally, a small snack was offered during the half-time break to maintain the energy and focus of the participants.



(a) (b)

Figure 3: Examples of target, laser dot, and visual feedback based on accuracy (numbers 11 and 4 in respective boxes). Note that the feedback text is perceived as larger and more readable when viewed through the headset. (a) Laser dot in center. (b) Laser dot in the fourth ring.

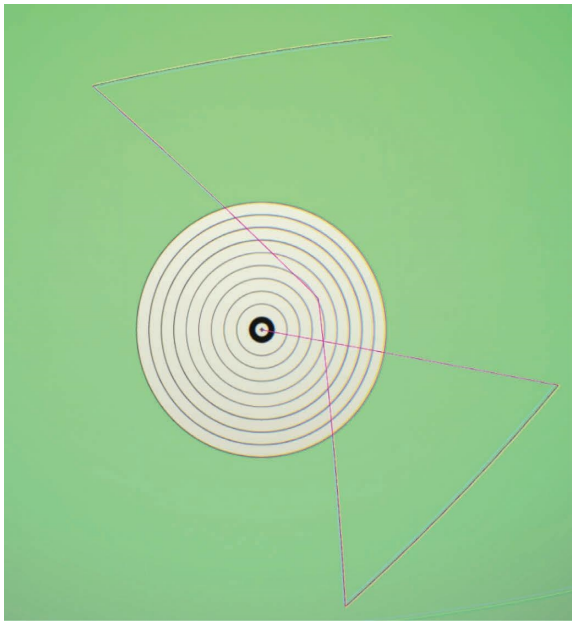


Figure 4: Example of an entire run with a path shown. The line length is based on the target speed that will be applied during that segment. Note that users will never be shown all lines during the experiment as seen here, but only the next three. In the no-path setting, no line is shown but the underlying motion logic is the same.

A Simulator-Sickness Questionnaire (SSQ) [60] was filled in by each participant before and after the experiment to determine whether the experiment program may be labelled a "problem simulator" [61] and thus potentially affect the results. The results of the SSQ are presented in Section V–H.

Swedish law requires an official ethical vetting on research that involves, for example, health risks or animal testing. The law can be applicable to VR experiments that are designed with the purpose of affecting research persons mentally or if the experiment poses an obvious risk of harm.

Table 1: Experiment parameters.

Parameter	Unit	Data							
Render path	Bool	0	1						
Added latency	Frames	0	1	2	4	8	16	32	64
Speed	m/s	0.25	0.5	1	2	4	8		
Angular speed μ^*	°/s	1.8	3.5	6.8	12.7	24.1	37.5		
Angular speed σ^*	°/s	0.1	0.2	0.6	1.5	3.2	9.7		

*Measured angular speed at the corresponding m/s, for reference, not a fixed parameter.

In this case, we found no such purpose and no risk of harm. An ethical approval was therefore not submitted.

5. Results and Discussion

A large amount of data was collected during experimentation and the most relevant results are presented here. Several plots were compiled and are shown with accompanying discussions. A synopsis of the chapter follows:

(1) Total Score: provides an overview of the variation in performance among participants.

(2) Average Accuracy: shows how the average accuracy, in terms of degrees off-center of the target, varies with latency, speed, and predictability of the path.

(3) Trends: shows how the average accuracy deteriorates with increasing latency depending on the path parameter.

(4) Average Accuracy over Time: shows the distinct average accuracy curves over time during each individual line movement.

(5) Peak Inaccuracy: shows the reaction times after target direction change as a function of latency.

(6) Optimal Target Sizes: mentions briefly how recommended target sizes can be derived from the data tables.

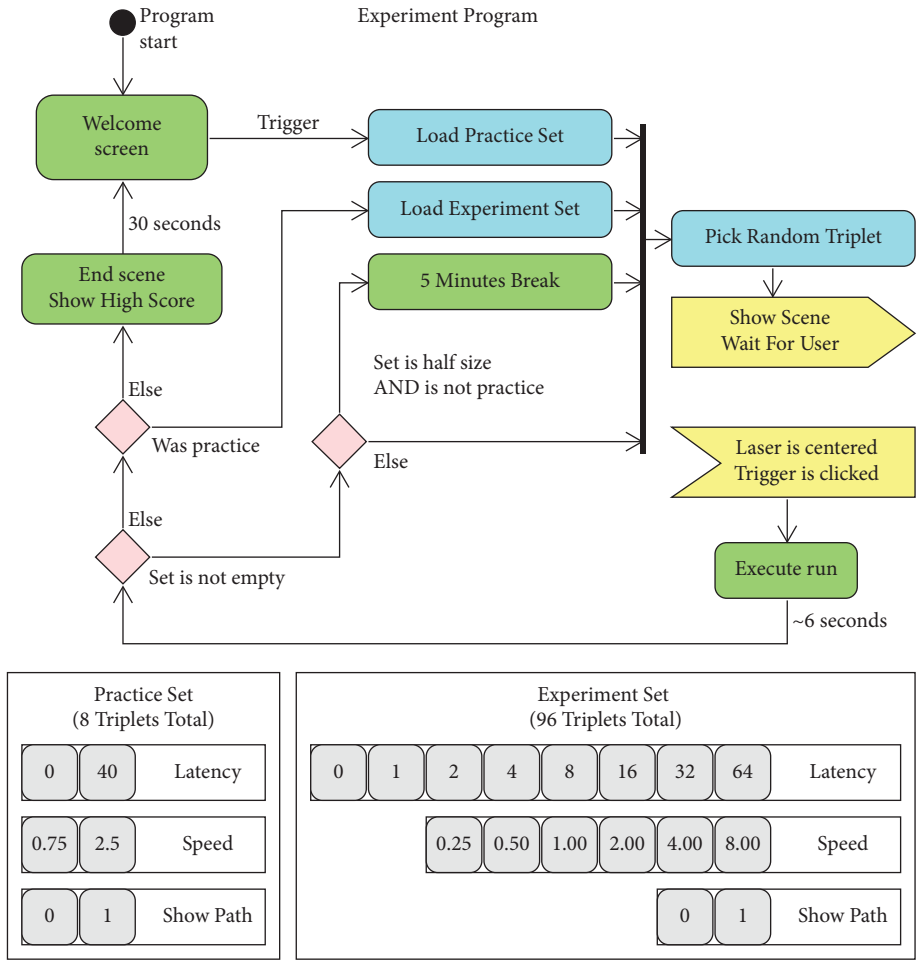


Figure 5: Overview of the experiment program. In short, a practice set of eight conditions is followed by 48 live conditions, followed by a pause and another 48.

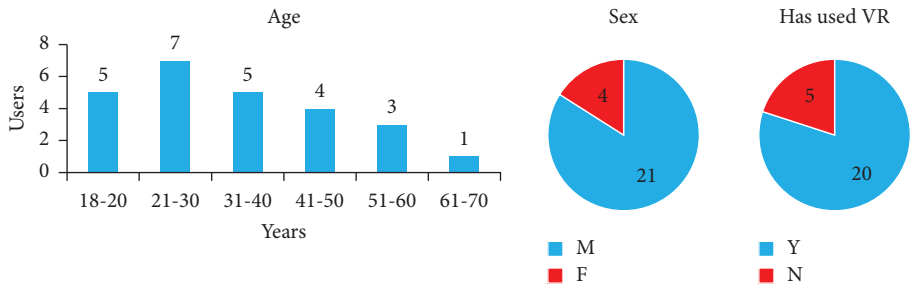


Figure 6: Demographic data of the 25 participants.

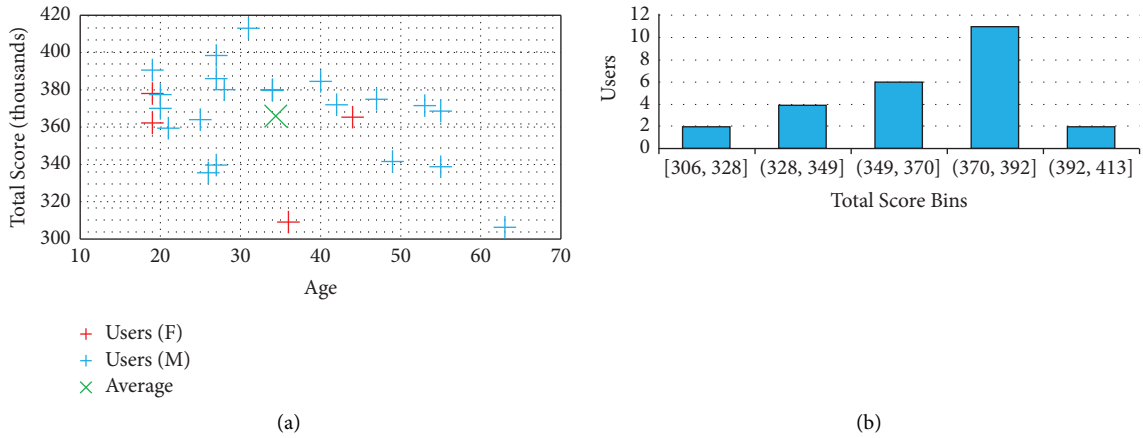
(7) Statistical Significance: shows t-tests applied to the collected data from which conclusions can be drawn.

(8) SSQ: shows the results of the SSQ.

5.1. Total Score. The participants gathered a total score throughout the experiment based on their performance in pointing the laser at the moving target. The score is calculated each frame based on the angle between the target center and laser dot. The maximum score is eleven and decreases by one in discrete steps shown as rings on the target model (see Figure 3).

Total scores ranged from 306 k to 413 k with mean $\mu = 366$ k and standard deviation $\sigma = 25$ k. To provide an overview, the scores of all individuals are illustrated in Figure 7.

The main take-away of this data follows: (1) The total score does not follow a Gaussian distribution at this number of samples. (2) There is no significant difference between the demographic groups in terms of total score. (3) There are



(a) (b)

Figure 7: Total score data. (a) The individual scores achieved by each participant. Min = 306 k, Max = 413 k, $\mu = 366$ k, $\sigma = 25$ k, $n = 25$. (b) The distribution of the total score (in thousands).

three borderline cases of outliers in the highest score and the two lowest scores.

Indeed, borderline cases may be considered outliers depending on the method of detection. For example, outliers may be detected by using interquartile ranges (IQR) [62] as follows with the outlier score k:

$$\begin{aligned} k &< \mathrm{Q1} - 1.5 \times \mathrm{IQR}, \\ k &> \mathrm{Q3} + 1.5 \times \mathrm{IQR}, \\ \mathrm{IQR} &= \mathrm{Q3} - \mathrm{Q1}. \end{aligned} \tag{1}$$

If the quartiles $Q1$ and $Q3$ are calculated inclusive of the median, where, e.g., $Q1$ is defined as the middle number in the lower half including the median, the three borderline cases are considered outliers. However, if $Q1$ and $Q3$ are calculated exclusive of the median, there are no outliers. Another simple method for detecting outliers is to define the upper and lower limits as $\mu \pm 3\sigma$, also in that case there are no outliers. Thus, the borderline cases are not extreme outliers but may or may not be included depending on the method and were therefore kept in the analysis in this work.

5.2. Average Accuracy. The total score shows the overall performance of the participants, while it is an arbitrary measure based on the target size in the 3D scene. A more generalized measure of accuracy is the angle between the target center and laser dot; this data is presented in Figure 8. The plots reveal that there is an increasingly significant difference in average accuracy between the path modes at higher latencies and speeds.

Showing the path allows for prediction, which explains the significantly better accuracy as speeds and latency increase. When speeds and latency are low, there is no significant difference which indicates that prediction may be unnecessary in those cases.

Another observation from the plots is the linear relationship in accuracy between higher speeds and latency levels. For example, the speeds 8, 4, and 2 m/s all yield similar average accuracies at 16, 32, and 64 frames of added latency, respectively. Thus, halving the speed allows for similar accuracy at double the latency, and vice versa.

5.3. Error-Increase Trends. It is evident that accuracy deteriorates with higher speeds and latencies and that there exists a threshold beyond 4 or 8 frames depending on path visibility where a significant increase in the average degree of error can be observed. The decrease in accuracy is clear when observed as an error-percentage increase between each latency step and this is plotted in Figure 9. While the increase varies irregularly when viewed at the individual speed-levels, plotting the average increase in error among all speed-levels reveals the underlying characteristic. A roughly linear increase is observed in the range 8–64 frames of latency when users can predict the target motion (show path), and a downward polynomial function can be observed in the no path condition at latencies 4–64 frames.

When showing the path, a linear increase in error with increased latency is an intuitive pattern. Because the user will be able to predict where the target will go and therefore follow it accordingly, but lag behind based on the added latency. It should be noted though, that there were a few exceptions to this rule where some users skipped the current line, due to it moving too fast for the given latency, and moved to the next line in order to get back on track with the pursuit of the target.

In the case of no path, the increase in average error between latency levels is not as steep, and the increase will wear off at the 64-frame mark. At 64 frames of latency, the increase is just 80%, as compared to 70% at 32 frames of latency. Thus, the increase in error wears off compared to show path where 64 frames of latency yield an increase of 110% from 32 frames, which increased by 60% from the lower level of 16 frames latency. This behaviour may be caused by limitations in the experiment. For example, the error cannot grow infinitely since the tested 3D-space is limited and the individual path lines always change direction after approximately one second. The accuracy may therefore reach a point where it cannot get much worse.

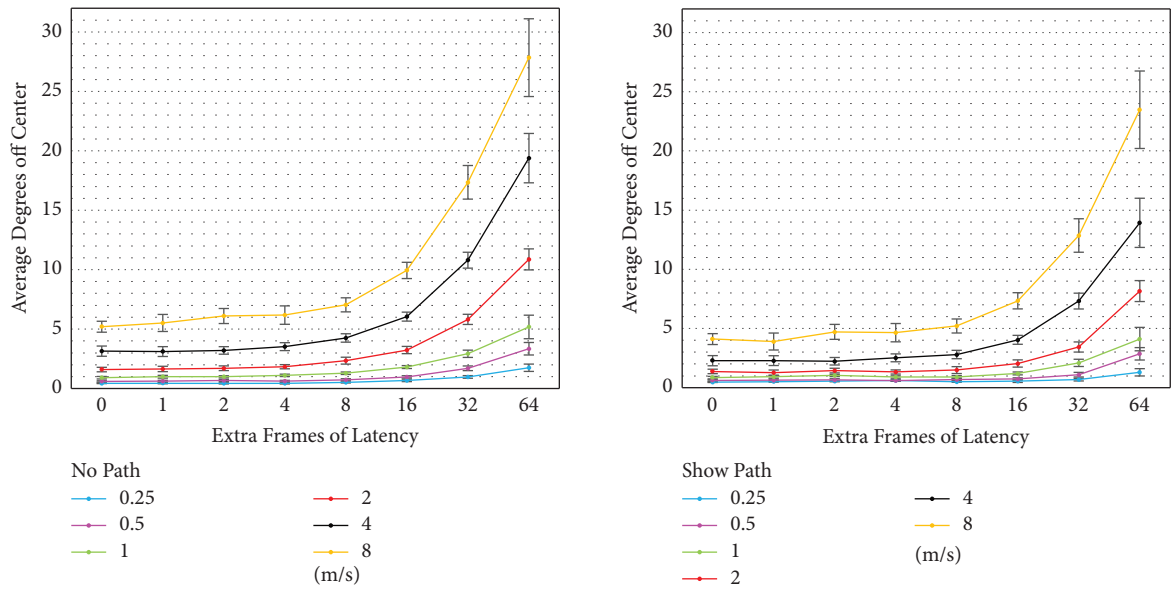


Figure 8: Average accuracy error plotted with latency on X for the different speeds (m/s) as series. (samples $n = 25$, significance level $\alpha = 0.05$).

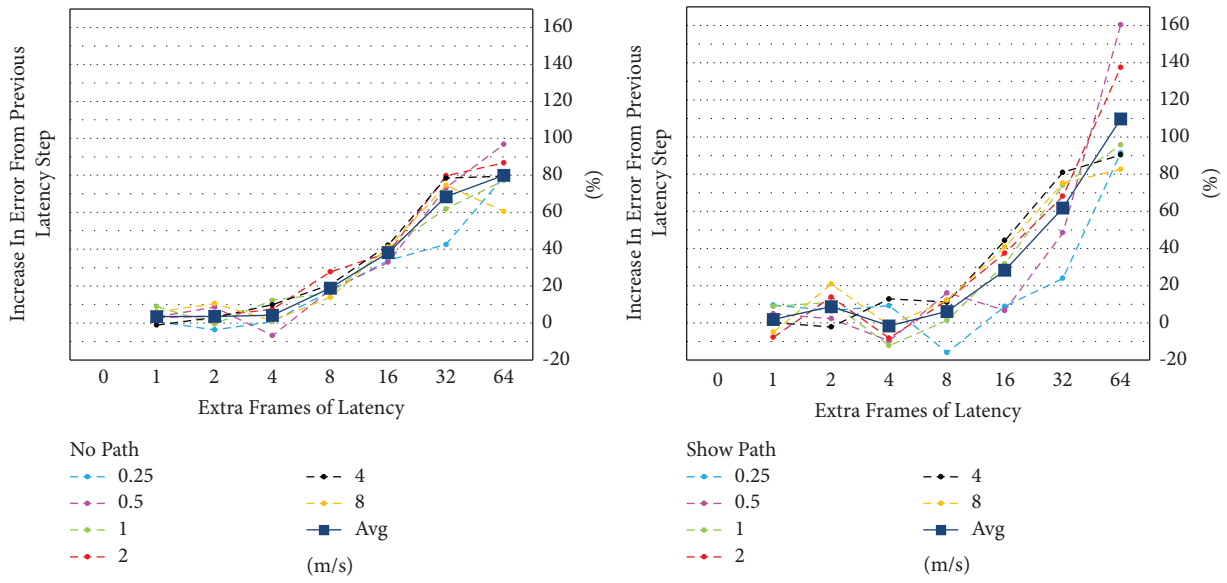


Figure 9: Increase in average error in each step of increased latency.

The main takeaways from Figure 9 are the two different thresholds at which the error starts to significantly increase and the difference in uniformity between the speed levels when a path is shown and when it is not.

5.4. Average Accuracy over Time. As the target moves in the experiment program at a particular speed and latency, it changes direction five times and runs for approximately 520 frames, which is almost 6 seconds at 90 fps. This means that a direction will be maintained for around 105 frames. A question that emerges is how long it takes for users to get back on point when these changes occur, because there will be a sudden change that the user must react to. To get insight into the latency of the readjustment period, the average error of each 105-frame run is plotted in Figure 10. The plots reveal how latency affects the average performance during the readjustment periods and in particular the different characteristics that occur when showing and not showing the upcoming target path during the experiment.

When a path is shown, the performance is more uniform throughout the run, and the peaks occur seemingly at random at lower speeds. The highest latency of 64 frames is an exception which reveals a distinct peak of inaccuracy at all speeds even in the Show Path case. At higher speeds, lower latencies also start to reveal peak inaccuracies visible by the concave downward curves. In contrast, when no path is shown, all speeds yield a distinct peak inaccuracy and

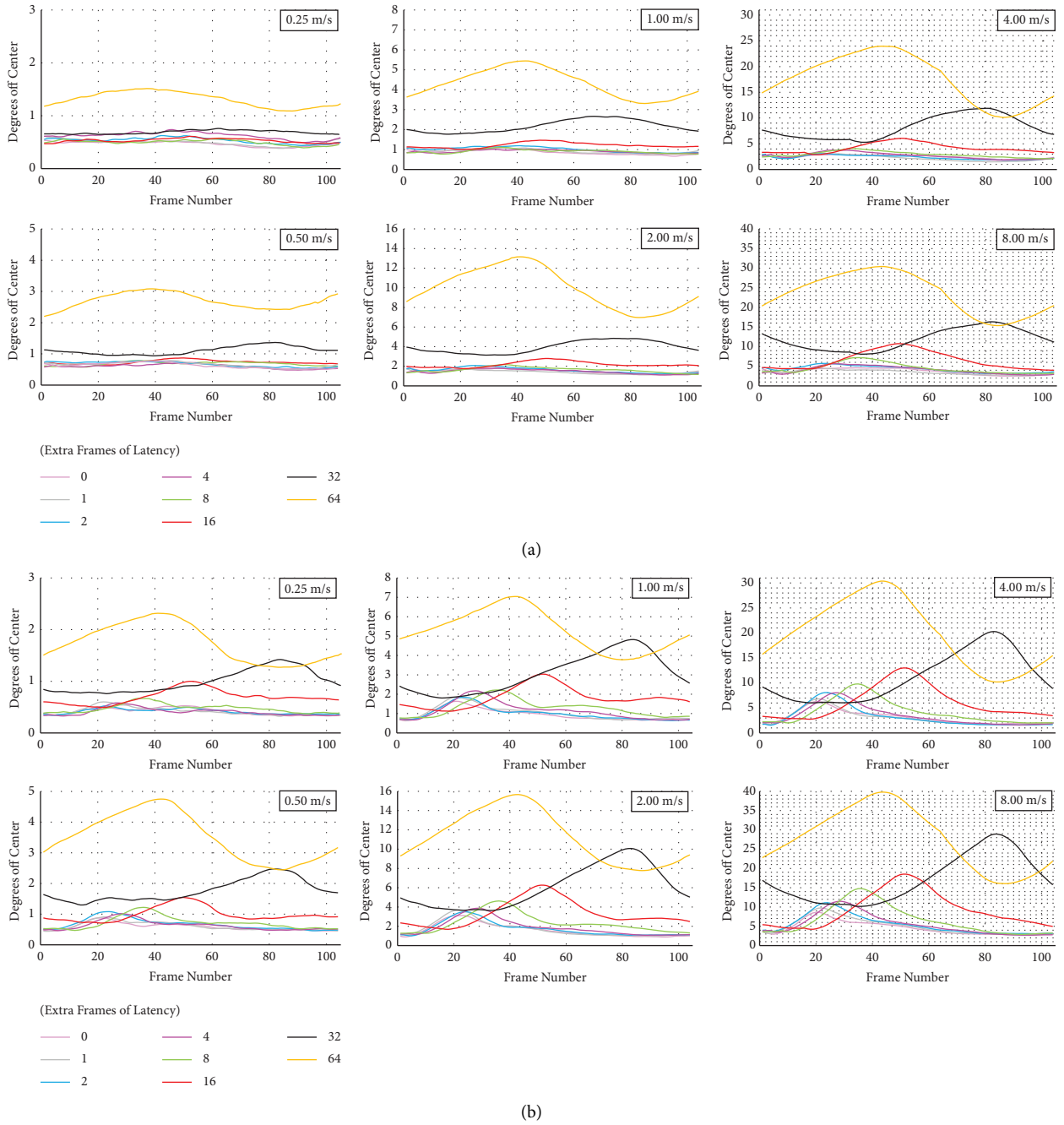


Figure 10: Average error curves of all experiments after the initial target direction has changed (the first line is excluded since the target is stationary at that starting point). The plotted data in the two subfigures are separated by the path parameter. (a) Show path. (b) No path.

downward concave curve characteristic, as can be observed in Figure 10.

An anomaly in the graphs is the widely different behaviour of the 64-frame curve when compared to all others. It begins with an earlier peak as compared to both the 32- and 16-frame curves and ends with an increasing error in every scenario contrary to all other latencies. It is not clear why this occurs, but it is possible that the users are always overshooting at this latency and do not have enough time to establish a stable pursuit of the target before it changes direction. If the target path lines were longer before changing direction, it is possible that the curve would reduce its amplitude over time and converge towards a minimal error. Conducting such an experiment would be part of future work though.

5.5. Peak Inaccuracy. To get a detailed understanding of how latency and target speed affect the readjustment delay, two plots were created that show the average peak error at all speeds and latencies (see Figure 11). These plots reveal that the peaks are random at lower latencies and speeds when a path is shown but remarkably similar for all speeds when

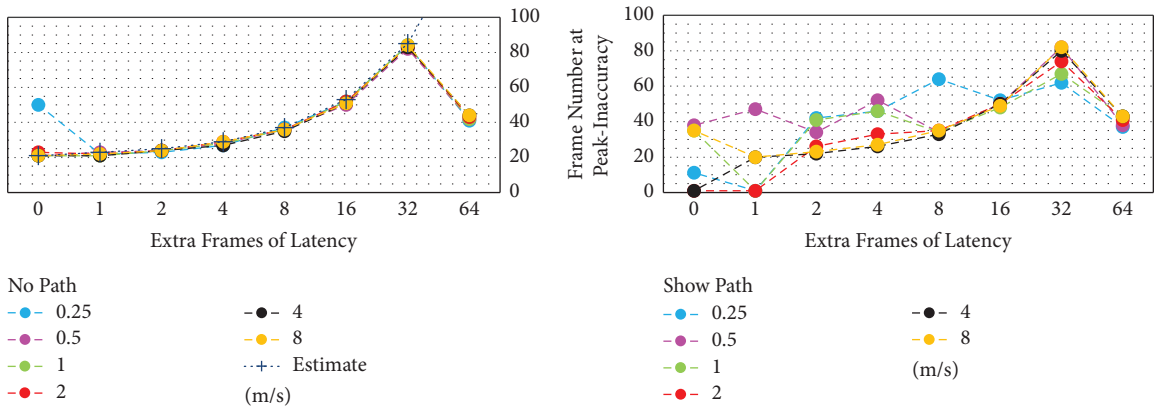


Figure 11: Peak inaccuracy index for all parameters. Estimate function for no path (left): $y = 21 + 2x$.

a path is not shown. This suggests that the readjustment latency is independent of speed when the path cannot be predicted; the absolute angular error is naturally higher at higher speeds though. Indeed, the average peak inaccuracy index between speeds can be estimated with $R^2 = 0.99$ by the simple linear function $y = 2x + 21$ shown in Figure 11 (excluding latency 64 and the outlier at latency 0 and speed 0.25 m/s). The derived function indicates that the average minimal reaction latency is 21 frames and that each added frame of latency adds two frames to this reaction time. 21 frames of latency are approximately equal to 235 ms at 89.53 fps. 235 ms is a reaction time in line with previous research that found the visual reaction time in medical students to be in approximately the range 220–250 ms [63]. However, recall that the inherent device latency was determined to be in the range 3-4 frames. Thus, the final result is inclusive of this delay which lowers the actual human reaction time to around 17–18 frames or roughly 190–200 ms. According to a literature review on human reaction times [64], the reported average simple reaction times to visual stimuli varies in literature between 180 and 220 ms, which is in line with the results. Simple reaction times are based on a single stimulus with a single possible response. Two other types of reaction-time experiments are also defined in the literature as recognition and choice experiments. In recognition experiments, there are some stimuli that should elicit a reaction and some that should not. In choice experiments, the stimuli dictate the correct response, e.g., showing the letter "A" should be responded to with a button press "A." The literature review found that recognition experiments in literature yielded average latencies of 384 ms and choice experiments 420–630 ms, depending on the number of choices [64]. Thus, it is clear that the conducted experiment falls into the category of simple reaction-time experiments, even though there are multiple possible directions to choose from.

5.6. Optimal Target Sizes. Numerical data in the form of the minimal, maximal, and average error as well as the peak inaccuracy index are provided in Tables 2 and 3. From Tables 2 and 3, recommended target sizes based on speed and latency can be derived. For example, to design for maximum accuracy at 64 additional frames of latency, a target moving and changing direction at 0.25 m/s at 7.5 m (approximately 1.8°/s) should be of the size 2.3° (no Path max), and an average accuracy of 1.3° can be expected at best (if the path is known, otherwise 1.7°).

5.7. Statistical Significance

5.7.1. Show/No Path. Paired two-sample t-tests were conducted between the average accuracy for all parameters between the two path modes. The resulting p values are presented in Table 4. Overall, the information is similar to the visual representation shown in Figure 8 but in a numerical form.

Since the number of t-tests is large, there is an increasing risk of randomly getting significantly small p values. The Bonferroni correction [65] suggests dividing the significance level by the number of tests $0.05/48 \approx 0.001$. Thus, p values above 0.001 in the tables presented in this section should be treated with some scepticism and are therefore marked in yellow, while p values above 0.05 are marked in red. The main takeaway of the table is how showing a path, and therefore providing predictability of target motion, becomes more important for accuracy as speed and latency increase. An exception is again the 64-frame case which indicates more randomness, possibly due to its difficulty regardless of path visibility and the mentioned strategy where some users skipped lines at high latencies to catch up with the target.

5.7.2. Latency. Additional t-tests were conducted on the latency parameter in order to clearly identify which latency levels significantly impair the accuracy. These tests compare the means of the 0-frame latency levels with higher latencies at the same speeds; they are presented in Table 5 for no path and VI for show path. The main take-away of these tables is the clear difference at the 8- and 16-frame latency levels. No path yields significantly worse accuracy for most speeds at 8 frames of latency while the limit is 16 frames for Show Path. Note also that there appears to be a transition phase towards lower p values at higher speeds already at the previous latency steps. 4 and 8 m/s at 8 frames of latency yield $p = 0.002$

Table 2: Accuracy statistics (no path).

	0.25				0.50				1.00				2.00				4.00				8.00			
	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p
64	1.3	2.3	1.7	41	2.4	4.8	3.5	42	3.8	7.0	5.3	42	7.8	15.7	11.3	43	10.2	30.4	19.9	44	16.1	39.8	27.8	44
32	0.8	1.4	1.0	84	1.3	2.5	1.8	82	1.8	4.8	3.0	84	3.6	10.1	6.0	83	6.0	20.3	11.1	83	10.2	28.9	17.1	84
16	0.5	1.0	0.7	53	0.7	1.5	1.0	50	1.1	3.0	1.8	52	1.7	6.3	3.3	52	2.7	13.0	6.1	51	4.2	18.5	9.3	51
8	0.4	0.7	0.5	37	0.5	1.2	0.7	36	0.8	2.2	1.3	36	1.3	4.6	2.4	36	2.0	9.8	4.2	35	2.9	14.8	6.6	36
4	0.3	0.6	0.4	27	0.5	1.0	0.6	28	0.7	2.2	1.1	28	1.1	3.9	1.9	29	1.6	8.0	3.4	27	2.7	11.5	5.5	29
2	0.3	0.5	0.4	23	0.5	1.1	0.7	24	0.6	1.8	1.0	24	1.0	3.5	1.7	24	1.6	8.1	3.2	24	3.1	11.1	5.4	24
1	0.3	0.6	0.4	22	0.4	0.9	0.6	22	0.7	1.9	1.0	21	1.0	3.7	1.7	22	1.6	6.8	3.0	21	2.8	9.8	4.9	22
0	0.3	0.5	0.4	50	0.4	0.9	0.6	21	0.6	1.6	0.9	21	0.9	3.1	1.6	23	1.7	6.3	3.1	21	2.9	8.6	4.5	21

∧, Min avg. error (°); ∨, max avg. error (°); μ, mean error (°); k_p, frame index at peak error (frame).

Table 3: Accuracy statistics (show path).

	0.25				0.50				1.00				2.00				4.00				8.00			
	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p	∧	∨	μ	k_p
64	1.1	1.5	1.3	37	2.2	3.1	2.7	38	3.3	5.4	4.3	43	7.0	13.2	10.0	41	10.1	24.0	17.6	43	15.4	30.4	23.5	43
32	0.6	0.8	0.7	62	0.9	1.4	1.1	82	1.8	2.7	2.2	67	3.1	4.8	4.0	74	5.3	11.9	8.2	80	8.1	16.3	11.9	82
16	0.5	0.6	0.5	52	0.6	0.9	0.7	50	1.0	1.5	1.2	48	1.8	2.8	2.2	50	2.9	6.1	4.2	50	4.0	10.8	6.5	49
8	0.4	0.6	0.5	64	0.6	0.8	0.7	34	0.8	1.1	0.9	34	1.2	2.2	1.6	35	2.1	4.1	2.9	33	3.2	7.2	4.5	35
4	0.5	0.7	0.6	46	0.5	0.7	0.6	52	0.8	1.0	0.9	46	1.1	1.8	1.5	33	1.8	3.7	2.7	26	2.8	5.7	4.0	27
2	0.4	0.6	0.5	42	0.5	0.8	0.7	34	0.8	1.2	1.0	41	1.2	2.1	1.6	26	1.8	3.1	2.3	22	3.2	5.7	4.3	23
1	0.4	0.6	0.5	1	0.5	0.8	0.7	47	0.8	1.1	0.9	1	1.1	1.7	1.4	1	1.5	3.2	2.2	20	2.6	4.6	3.5	20
0	0.4	0.5	0.5	11	0.5	0.8	0.6	38	0.7	1.1	0.9	35	1.1	1.9	1.5	1	1.9	3.0	2.5	1	2.5	4.6	3.7	35

∧, Min avg. error (°); ∨, max avg. error (°); μ, mean error (°); k_p, frame index at peak error (frame).

Table 4: P values of paired two sample T-test (means of no path compared to show path).

	0	1	2	4	8	16	32	64
0.25	0.386	0.114	0.034	0.016	0.826	0.016	≤0.001	0.033
0.50	0.588	0.503	0.883	0.495	0.239	≤0.001	≤0.001	0.426
1.00	0.228	0.161	0.801	0.002	≤0.001	≤0.001	≤0.001	0.049
2.00	0.053	≤0.001	0.014	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001
4.00	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001
8.00	≤0.001	≤0.001	0.042	0.001	≤0.001	≤0.001	≤0.001	0.039

Table 5: P values of paired two sample T-test (means compared to zero latency, no path).

	0	1	2	4	8	16	32	64
0.25		0.845	0.584	0.661	0.001	≤0.001	≤0.001	≤0.001
0.5		0.406	0.022	0.162	≤0.001	≤0.001	≤0.001	≤0.001
1		0.010	0.045	≤0.001	≤0.001	≤0.001	≤0.001	≤0.001
2		0.370	0.167	0.001	≤0.001	≤0.001	≤0.001	≤0.001
4		0.783	0.653	0.041	≤0.001	≤0.001	≤0.001	≤0.001
8		0.254	0.007	0.004	≤0.001	≤0.001	≤0.001	≤0.001

for show path and, for no path, 1, 2, 4, and 8 m/s at 4 frames of latency yield $p \leq 0.001$, $p = 0.001$, $p = 0.041$, and $p = 0.004$, respectively.

5.8. SSQ. The SSQ [60] was filled out by the participants before and after the experiment in order to determine whether the experiment procedure may trigger potentially performance-

Table 6: P values of paired two sample T-test (means compared to zero latency, show path).

	0	1	2	4	8	16	32	64
0.25		0.202	0.054	0.048	0.245	0.035	≤0.001	≤0.001
0.5		0.576	0.393	0.536	0.106	0.012	≤0.001	0.005
1		0.255	0.264	0.471	0.211	≤0.001	≤0.001	≤0.001
2		0.341	0.539	0.604	0.325	≤0.001	≤0.001	≤0.001
4		0.979	0.691	0.178	0.002	≤0.001	≤0.001	≤0.001
8		0.469	0.321	0.120	0.002	≤0.001	≤0.001	≤0.001

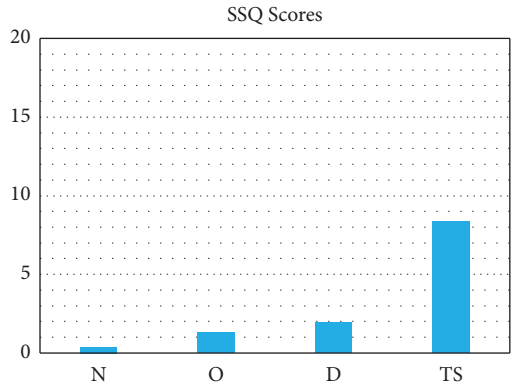


Figure 12: The SSQ scores of the experiment: nausea (N), oculomotor (O), disorientation (D), and total score (TS).

degrading symptoms. A simulator is said to be a "problem simulator" if one of the scores reaches beyond 20 [61]. The compiled score of the SSQ-data is presented in Figure 12 and does not indicate problematic levels in any of the scores.

The time spent in VR during the experiment session was relatively long (around 10–15 min). The half-time pause of five minutes and the offering of a small snack may have helped in maintaining the energy of participants and avoiding any potentially strong negative symptoms covered by the SSQ. Note that the participants were informed that they could end the experiment at any time. Yet, all participants chose to complete it.

6. Conclusions

Human performance in terms of hand-controller accuracy in VR has been measured with the varying parameters latency, target speed, and predictability of target path. The tests have been carried out in a context where a target changes direction multiple times while moving in straight lines. The collected data have been presented and the main conclusions given in tested context is as follows:

(1) Predictability significantly improves the average accuracy at higher speeds and latencies. Generally, as indicated by Table 4, prediction becomes significantly important ($\alpha = 0.05$) at speeds beyond approximately 3.5°/s (1 m/s at 7.5 m) and at latencies beyond roughly 130 ms (8 + 4 frames at 90 fps, where 4 is the inherent device latency).

(2) Latency has a significantly higher impact on accuracy when the target path cannot be predicted. Tables 5 and 6 indicate that scores start to get significantly worse with No Path at latencies of 4 + 4 frames (≈ 90 ms) with a speed of 1 m/s (≈ 6.8°/s). Show path yielded significantly worse scores starting later, at 16 + 4 frames (≈ 220 ms) of latency and 1 m/s (≈ 6.8°/s) target speed.

(3) When the target changes to a direction that cannot be predicted, the average peak inaccuracy occurs on average 190–200 ms later (excluding device latency), which is in line with related work on simple reaction times with a single stimulus and response [64].

(4) As indicated in Figure 11, the frame index of the peak inaccuracy after direction change increases by latency multiplied by 2 when the new target direction cannot be predicted.

7. Limitations and Future Work

7.1. Game Contexts. In future work, the data and insights generated from this study may be tested in praxis-relevant scenarios, for example, in remote VR game contexts containing some aiming components. We hypothesize that the given latencies, motion speeds, and corresponding accuracy levels are applicable directly and/or scale with similar characteristics in other 3D scenes. Also, we expect that the results based on predictability are applicable to scenarios with no visible target path but that nonetheless contain a predictable motion path that must be followed according to the rules of the given scenario. The predictability of such paths may be communicated to the user by rendering them directly as a helping overlay, as was done in this study, but it can also be communicated more subtly by other means. To maintain the realism of the 3D content, the guidance that conveys upcoming motions can be integrated into the specific game scene. Trivial examples could be trains that are part of the game and act as targets in some manner; they move along rails, clearly visible to the user, without requiring an unnatural overlay. Cars are another example, moving along roads. Inertia is a property that also can be used to convey information about motion; for example, slowly moving naval vessels in

a war-themed game may be more suitable targets to users with high input latency than erratically moving airborne drones. More complex objects are characters, whether human or otherwise. They are common in games and typically need to be able to move relatively freely. Hinting where such characters are heading next is not trivial, but it is possible to give some indication by using animations. For example, characters about to make a jump may first need to play an animation that shows the bracing before the jump, and movement on the ground can be driven by the animation of the legs. Another scenario is sports-themed games, which typically contain fast-moving airborne items, such as tennis balls or hockey pucks. While such projectiles will follow a predictable path according to physics, their change in direction when, for example, another player hits them, remains difficult to predict, and is typically a central part of the game. In such scenarios, where prediction is not possible and/or unwanted, one may consider including adjustable parameters for the target speed and size instead, if accommodation for high input latency is a priority.

Adding the predictability of motions, lowering the target speeds, and/or increasing the target sizes reduces or mitigates the negative effects on accuracy as latency increases. For instance, the study indicates, based on Table 2, that a target without a path, moving at 4 m/s ($\approx 24.1°$/s), yields an accuracy within 3.1° on average without additional input latency. That size almost doubled, to 6.1°, at 16 frames of additional latency. Indeed, there is an approximate doubling at all target speeds for these latency levels when no path is shown. Future research may perform experiments with these numbers in game contexts to determine whether the data is sufficiently accurate in practice. For example, can users with 16 additional frames of input latency in the HTC Vive play, e.g., a ping-pong game and perform similarly as without extra latency if their ping-pong balls are doubled in size?

The study indicates that it is best to keep the total input latency below ≈ 90 ms. If that cannot be done, the game design may alleviate the negative effects on accuracy of remote operation in various ways by adjusting target parameters and providing hints about upcoming motions. However, it is still of critical importance, since the context is entertainment, that this does not negatively impact the "fun-factor" of the game. We may be able to hit slowly moving, large balls in high-latency ping-pong, but is it still a fun game? This is an important question that falls outside the scope of this study as only the objective performance was recorded and analyzed. Indeed, one may be able to adapt to latency and maintain the performance, but it may make the task increasingly annoying to perform, which is unsuitable when the purpose is entertainment.

7.2. Physical Actuation Contexts. Outside entertainment, there are other contexts where a path should be followed while input may be delayed. We have considered remote surgery to be one such potential scenario. A simple example would be an operation in which a surgeon makes a cut through skin while using remote robotic tools, where the cut should follow a correct path with high accuracy. However, in terms of surgery, the applicability of our experimentation methods may be limited to simple examples. When operating inside the body or when otherwise performing complex motions such as stitching, the motion path is no longer dependent on just two dimensions, but three. The surgeon must not only cut at the correct X and Y coordinates but also at the correct depth and direction (Z). Accuracy measurements in 3D are outside the scope of this study and part of potential future research. An experiment measuring the accuracy in 3D could be conducted, for example, by rendering a form of cone shape along a target 3D path. The pointed edge of the cone would indicate from where and in what direction the controller should be pointing. To construct the correct 3D path, one could, for example, record the operator motions while the task of interest is performed accurately without additional latency. The 3D path may then be played back in a simulator where an experiment conducting person tries to follow this path as accurately as possible while latency is injected into the controller input. It may then be possible to determine at which level of latency the task can be performed accurately, and running the experiment could even be useful for training. Still, it is not evident that this method would provide accurate results for complex tasks such as surgery. The complex task may involve multiple correct options and higher-level decision-making, and the speed is not fixed but can be decided by the operator.

Another potential scenario outside of entertainment is in military applications, where the remote feed may originate from a camera sight used for manually aiming some weapon. In that case, the study indicates the expected accuracy depending on input latency and target speed. However, one would also need to consider the projectile speed in that case, which is outside the scope of this study. Furthermore, the controller mechanics would likely be different from VR hand controllers, which may significantly impact the results.

Data Availability

The data and the experiment program are available at https://github.com/kelkka/VRControllerLatency.

Disclosure

A preprint of this article was included in the PhD thesis titled "Evaluation and Reduction of Temporal Issues in Remote VR" [66].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

The authors wish to thank the KK Foundation for the funding of the ViaTecH and HINTS projects under the contract nos. 20170056 and 20220068, respectively. A special

thanks also goes out to everyone who participated in the experiment.

References

[1] M. Abdallah, C. Griwodz, K.-T. Chen, G. Simon, P.-C. Wang, and C.-H. Hsu, "Delay-sensitive video computing in the cloud: a survey," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 14, no. 3s, pp. 1–29, 2018.

[2] S. Solotko, "The instantaneous cloud: emerging consumer applications of 5G wireless networks," 2018, https://www.tiriasresearch.com/downloads/the-instantaneous-cloud-emerging-consumer-applications-of-5g-wireless-networks/.

[3] M. Abrash, "What VR could, should, and almost certainly will be within two years," 2014, https://media.steampowered.com/apps/steamdevdays/slides/vrshouldbe.pdf.

[4] 3GPP, "Virtual reality (VR) media services over 3gpp (release 16)," Technical Report, 3GPP, Valbonne, France, 2018.

[5] Gsm Association, "Cloud AR/VR whitepaper," 2019, https://www.gsma.com/futurenetworks/wiki/cloud-ar-vr-whitepaper/.

[6] M. Abrash, "Latency-the sine qua non of AR and VR," 2012, http://blogs.valvesoftware.com/abrash/latency-the-sine-qua-non-of-ar-and-vr/.

[7] A. Seam, "Enabling mobile augmented and virtual reality with 5G networks," 2018, https://about.att.com/ecms/dam/snrdocs/Foundry.20ARVR.20Public.20Whitepaper.20.pdf.

[8] U. H. List, *Nonlinear Prediction of Head Movements for Helmet-Mounted Displays*, Air Force Human Resources Lab Brooks AFB TX, San Fransisco, SF, USA, 1983.

[9] R. M. Taylor II, "Virtual reality system concepts illustrated using OSVR," in *VR Developer Gems*, W. R. Sherman, Ed., CRC Press, Boca Raton, FL, USA, 2019.

[10] R. H. Y. So and M. J. Griffin, "Compensating lags in head-coupled displays using head position prediction and image deflection," *Journal of Aircraft*, vol. 29, no. 6, pp. 1064–1068, 1992.

[11] J. M. P. van Waveren, "The asynchronous time warp for virtual reality on consumer hardware," in *Proceedings of the ACM Conf. On Virtual Reality Software and Technology*, pp. 37–46, Munich, Germany, November 2016.

[12] J. Carmack, "Latency mitigation strategies," 2013, https://danluu.com/latency-mitigation/.

[13] W. R. Mark, L. McMillan, and G. Bishop, "Post-rendering 3D warping," in *Proceedings of the 1997 symposium on Interactive 3D graphics*, April 1997.

[14] F. A. Smit, R. van Liere, and B. Fröhlich, "The design and implementation of a VR-architecture for smooth motion," in *Proceedings of the ACM Symp. On Virtual Reality Software And Technology, Ser. VRST '07*, pp. 153–156, Association for Computing Machinery, New York, NY, USA, November 2007.

[15] E. M. Peek, B. C. Wünsche, and C. Lutteroth, "Image warping for enhancing consumer applications of head-mounted displays," in *Proceedings of the Australasian User Interface Conf*, pp. 47–55, Auckland, New Zealand, January 2014.

[16] Y.-C. Li, C.-H. Hsu, Y.-C. Lin, and C.-H. Hsu, "Performance measurements on a cloud vr gaming platform," in *Proceedings of the 1st Workshop on Quality of Experience (QoE) in Visual Multimedia Applications*, October 2020.

[17] N. Somraj, P. Sancheti, and R. Soundararajan, "Temporal view synthesis of dynamic scenes through 3D object motion estimation with multi-plane images," in *Proceedings of the IEEE Int. Symp. On Mixed and Augmented Reality (ISMAR)*, pp. 817–826, Singapore, October 2022.

[18] V. Kanchana, N. Somraj, S. Yadwad, and R. Soundararajan, "Revealing disocclusions in temporal view synthesis through infilling vector prediction," in *Proceedings of the IEEE/CVF Winter Conf. On Applications of Computer Vision (WACV)*, pp. 3093–3102, Waikoloa, HI, USA, January 2022.

[19] R. Held and N. Durlach, "Telepresence, time delay and adaptation," *Pictorial communication in virtual and real environments*, vol. 01, pp. 232–246, 1993.

[20] J. E. Conklin, "Effect of control lag on performance in a tracking task," *Journal of Experimental Psychology*, vol. 53, no. 4, pp. 261–268, 1957.

[21] G. L. Ricard, "Manual control with delays: a bibliography," *SIGGRAPH Comput. Graph*, vol. 28, no. 2, pp. 149–154, 1994.

[22] J. L. Adams, *An Investigation of the Effects of Time Lag Due to Long Transmission Distances upon Remote Control*, National Aeronautics and Space Administration, Washington, DC, USA, 1961.

[23] W. R. Ferrell, "Remote manipulation with transmission delay," *IEEE Transactions on Human Factors in Electronics*, vol. 6, no. 1, pp. 24–32, 1965.

[24] W. R. Corliss and E. G. Johnsen, "Teleoperator controls. an aec-nasa technology survey," 1968, https://www.osti.gov/biblio/4797359.

[25] F. R. Cooper, W. T. Harris, and V. J. Sharkey, *The Effect of Delay in the Presentation of Visual Information on Pilot Performance*, Naval Training Equipment Center, Orlando, FL, USA, 1975.

[26] G. K. Miller and D. R. Riley, *The Effect of Visual-Motion Time Delays on Pilot Perforamnce in a Simulated Pursuit Tracking Task*, NASA, Hampton, Virginia, 1977.

[27] R. H. Y. So and M. J. Griffin, "Effects of time delays on head tracking performance and the benefits of lag compensation by image deflection," in *Proceeding of the Flight Simulation Technologies Conference*, New Orleans, LS, USA, August 1991.

[28] B. Sorensen, M. Donath, G.-B. Yang, and R. Starr, "The Minnesota scanner: a prototype sensor for three-dimensional tracking of moving body segments," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 4, pp. 499–509, 1989.

[29] Z. Lai, Y. C. Hu, Y. Cui, L. Sun, N. Dai, and H.-S. Lee, "Furion: engineering high-quality immersive virtual reality on today's mobile devices," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1586–1602, 2020.

[30] E. Cuervo, A. Wolman, L. P. Cox et al., "Kahawai: high-quality mobile gaming using gpu offload," in *Proceedings of the Annual Int. Conf. On Mobile Systems, Applications, and Services*, pp. 121–135, Association for Computing Machinery, New York, NY, USA, May 2015.

[31] T. Kämäräinen, M. Siekkinen, J. Eerikäinen, and A. Ylä-Jääski, "Cloud accelerated interactive mobile virtual reality," in *Proceedings of the ACM Int. Conference On Multimedia*, pp. 1181–1189, Association for Computing Machinery, New York, NY, USA, October 2018.

[32] V. Kelkkanen, M. Fiedler, and D. Lindero, "Synchronous remote rendering for VR," *International Journal of Computer Games Technology*, vol. 2021, Article ID 6676644, 16 pages, 2021.

[33] K. Chung, J. Ji, and R. So, "Manual control with time delays in an immersive virtual environment," pp. 211–218, 2011, https://hdl.handle.net/1783.1/38530.

[34] K. Brunnström, E. Dima, M. Andersson, M. Sjöström, T. Qureshi, and M. Johanson, "Quality of experience of hand

controller latency in a virtual reality simulator," *Electronic Imaging*, vol. 31, no. 12, pp. 218-1–218-9, 2019.

[35] D. Thirtyacre, *The effects of remotely piloted aircraft command and control latency during within-visual-range air-to-air combat*, Ph.D. Dissertation, Embry-Riddle Aeronautical University, Daytona Beach, FL, USA, 2021.

[36] F. L. Luro and V. Sundstedt, "A comparative study of eye tracking and hand controller for aiming tasks in virtual reality," in *Proceedings of the ACM Symp. On Eye Tracking Research & Applications*, June 2019.

[37] R. Brown, "Htc vive: full specification-vrcompare," 2020, https://vr-compare.com/headset/htcvive.

[38] R. B. Miller, "Response time in man-computer conversational transactions," in *Proceedings of the Fall Joint Computer Conference, Part I, ser. AFIPS '68 (Fall, part I)*, pp. 267–277, Association for Computing Machinery, New York, NY, USA, December 1968.

[39] C. Attig, N. Rauh, T. Franke, and J. F. Krems, "System latency guidelines then and now – is zero latency really considered necessary?" in *Engineering Psychology and Cognitive Ergonomics: Cognition and Design*, D. Harris, Ed., Cham: Springer International Publishing, Salmon Tower Building NY, USA, pp. 3–14, 2017.

[40] C. Jay, M. Glencross, and R. Hubbold, "Modeling the effects of delayed haptic and visual feedback in a collaborative virtual environment," *ACM Transactions on Computer-Human Interaction*, vol. 14, no. 2, p. 8, 2007.

[41] C. Jay and R. Hubbold, "Delayed visual and haptic feedback in a reciprocal tapping task," in *Proceedings of the Joint Eurohaptics Conf. And Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 655-656, World Haptics Conference, New York, NY, USA, March 2005.

[42] M. Annett, A. Ng, P. Dietz, W. F. Bischof, and A. Gupta, "How low should we go? Understanding the perception of latency while inking," in *Proceedings of the Graphics Interface 2014*, pp. 167–174, CAN: Canadian Information Processing Society, Natick, MS, USA, November 2014.

[43] A. Ng, J. Lepinski, D. Wigdor, S. Sanders, and P. Dietz, "Designing for low-latency direct-touch input," in *Proceedings of the ACM Symp. On User Interface Software And Technology*, pp. 453–464, Association for Computing Machinery, New York, NY, USA, October 2012.

[44] J. Marescaux, J. Leroy, F. Rubino et al., "Transcontinental robot-assisted remote telesurgery: feasibility and potential applications," *Annals of Surgery*, vol. 235, no. 4, pp. 487–492, Apr. 2002.

[45] M. Anvari, C. McKinley, and H. Stein, "Establishment of the world's first telerobotic remote surgical service," *Annals of Surgery*, vol. 241, no. 3, pp. 460–464, Mar. 2005.

[46] J. R. Sterbis, E. J. Hanly, B. C. Herman et al., "Transcontinental telesurgical nephrectomy using the da vinci robot in a porcine model," *Urology*, vol. 71, no. 5, pp. 971–973, 2008.

[47] M. Perez, F. Quiaios, P. Andrivon et al., "Paradigms and experimental set-up for the determination of the acceptable delay in telesurgery," in *Proceedings of theInt. Conf. Of the IEEE Engineering in Medicine and Biology Society*, pp. 453–456, Lyon, France, August 2007.

[48] A. Kumcu, L. Vermeulen, S. A. Elprama et al., "Effect of video lag on laparoscopic surgery: correlation between performance and usability at low latencies," *International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 13, no. 2, Article ID e1758, 2017.

[49] P. Barba, J. Stramiello, E. K. Funk, F. Richter, M. C. Yip, and R. K. Orosco, "Remote telesurgery in humans: a systematic review," *Surgical Endoscopy*, vol. 36, no. 5, pp. 2771–2777, 2022.

[50] W. R. Sherman and A. B. Craig, "Chapter 4-input: interfacing the participant(s) with the virtual world," in *Ser. The Morgan Kaufmann Series in Computer Graphics*, W. R. Sherman and A. B. Craig, Eds., pp. 190–256, Morgan Kaufmann, Boston, MS, USA, 2nd edition, 2018.

[51] O. Kreylos, "Lighthouse tracking examined," 2016, http://doc-ok.org/?p=1478.

[52] M. Warburton, M. Mon-Williams, F. Mushtaq, and J. R. Morehead, "Measuring motion-to-photon latency for sensorimotor experiments with virtual reality systems," *Behavior Research Methods*, pp. 1554–3528, 2022.

[53] D. C. Niehorster, L. Li, and M. Lappe, "The accuracy and precision of position and orientation tracking in the HTC Vive virtual reality system for scientific research," *I-Perception*, vol. 8, no. 3, Article ID 204166951770820, 2017.

[54] J. A. Jones, E. Luckett, T. Key, and N. Newsome, "Latency measurement in head-mounted virtual environments," in *Proceedings of the IEEE Conf. On Virtual Reality and 3D User Interfaces (VR)*, Osaka, Japan, March 2019.

[55] M. L. Chénéchal and J. C. Goldman, "HTC Vive Pro Time performance benchmark for scientific research," in *Proc. Int. Conf. On Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*, G. Bruder, S. Yoshimoto, and S. Cobb, Eds., The Eurographics Association, Saarbrücken, Germany, 2018.

[56] J.-P. Stauffert, F. Niebling, and M. E. Latoschik, "Simultaneous run-time measurement of motion-to-photon latency and latency jitter," in *Proceedings of the IEEE Conf. On Virtual Reality and 3D User Interfaces (VR)*, pp. 636–644, Atlanta, GA, USA, March 2020.

[57] Valve, "OpenVR SDK," 2015, https://github.com/ValveSoftware/openvr.

[58] V. Kelkkanen and M. Fiedler, "A test-bed for studies of temporal data delivery issues in a tpcast wireless virtual reality set-up," in *Proceedings of the 28th International Telecommunication Networks and Applications Conference (ITNAC)*, November 2018.

[59] V. Kelkkanen, M. Fiedler, and D. Lindero, "Bitrate requirements of non-panoramic vr remote rendering," in *Proceedings of the 28th ACM International Conference on Multimedia*, Seattle, WA, USA, October 2020.

[60] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal, "Simulator sickness questionnaire: an enhanced method for quantifying simulator sickness," *The International Journal of Aviation Psychology*, vol. 3, no. 3, pp. 203–220, 1993.

[61] R. Kennedy, J. Drexler, D. Compton, K. Stanney, D. Lanham, and D. Harm, "Configural scoring of simulator sickness,

cybersickness and space adaptation syndrome: similarities and differences?," in *Virtual and Adaptive Environments: Applications, Implications, and Human Performance Issues*, L. J. Hettinger and M. W. Haas, Eds., p. 247, Lawrence Erlbaum Associates, Inc, Mahwah, NJ, USA, 2003.

[62] Wikipedia, "Interquartile range," 2023, https://en.wikipedia.org/wiki/Interquartile_range.

[63] A. Jain, R. Bansal, A. Kumar, and K. D. Singh, "A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students," *International Journal of Applied and Basic Medical Research*, vol. 5, no. 2, pp. 124–127, 2015.

[64] R. J. Kosinski, "A literature review on reaction time kinds of reaction time experiments," 2012, https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/biae.clemson.edu/bpc/bp/Lab/110/reaction.htm.

[65] C. Bonferroni, "Teoria statistica delle classi e calcolo delle probabilità, ser. pubblicazioni del r. istituto superiore di scienze economiche e commerciali di firenze. seeber," 1936, https://books.google.se/books?id=3CY-HQAACAAJ.

[66] V. Kelkkanen, *Evaluation and reduction of temporal issues in remote vr*, Ph.D. Dissertation, Blekinge Institute of Technology, Karlskrona, Sweden, 2023.