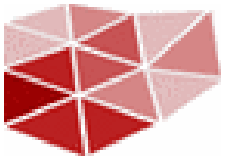


Dynamic Control of Interocular Distance and Vergence Angle in Rendering for Head-Mounted Stereo Displays

Jochen Jacobs
(373163)

Bachelor Thesis

Fakultät IV - Elektrotechnik und Informatik
Technischen Universität Berlin

05.07.2019

Supervisor : Prof. Dr. Marc Alexa
Prof. Dr.-Ing. Olaf Hellwich
Advisor : Xi Wang

Eidesstattliche Versicherung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Die selbständige und eigenständige Anfertigung versichert an Eides statt:

Berlin, den 05. Juli 2019 .....

Jochen Jacobs

Dynamic Control of Interocular Distance and Vergence Angle in Rendering for Head-Mounted Stereo Displays

Abstract In recent years, head-mounted displays have become increasingly popular. However, the well known vergence-accommodation conflict causes significant amount of discomfort. Some research has proposed to dynamically adjust the rendering to change the eye vergence needed for the currently focused object to match the accommodation.

This thesis uses camera vergence and separation to change the eye vergence based on the users gaze. To determine the gaze depth, a probabilistic algorithm is proposed that uses both measured eye-convergence and scene geometry. An experiment is conducted to determine whether the dynamic vergence and separation changes reduce fatigue. Fatigue is measured objectively (pupil diameter, pupil diameter variance, eye movement speed, blink rate, and reaction time) and subjectively using a questionnaire. The experiment is also used to evaluate the quality of the depth estimation algorithm.

Results show, that the output of the proposed algorithm to estimate the eye vergence depth is better than simple raycasting and ray intersection algorithms. However the objective evaluation of the adjustment method shows that it does not help reduce fatigue but in fact increases discomfort. The subjective evaluation was inconclusive but trends go into the same direction.

Keywords head-mounted displays, vergence, fatigue, eye-tracking

Dynamische Steuerung von Kameraabstand und Vergenz-Winkel beim Rendern für Head-Mounted Sterodisplays

Zusammenfassung In den letzten Jahren haben sich Headmounted Displays immer mehr durchgesetzt. Der bekannte *vergence-accommodation* Konflikt verursacht jedoch erhebliches Unwohlsein. Einige Forschungen haben vorgeschlagen, das Rendering dynamisch anzupassen, um die Augenkonvergenz zu ändern, die für das aktuell fokussierte Objekt erforderlich ist und diese mit der Fokusierung der Augen in Einklang zu bringen.

Diese Arbeit verwendet Kamera-Vergenz und Abstand, um die Augenvergenz basierend auf der Blicktiefe des Benutzers zu ändern. Um die Blicktiefe zu bestimmen, wird ein probabilistischer Algorithmus vorgeschlagen, der sowohl gemessene Blickkonvergenz als auch Szenengeometrie verwendet. Es wird ein Experiment durchgeführt, um festzustellen, ob die dynamischen Vergenz- und Abstandänderungen die Augenermüdung reduzieren. Die Ermüdung wird objektiv (Pupillendurchmesser, Pupillendurchmesservarianz, Augenbewegungsgeschwindigkeit, Blinzelrate und Reaktionszeit) und subjektiv mittels eines Fragebogens gemessen. Das Experiment wird auch verwendet, um die Qualität des Tiefenschätzungsalgorithmus zu bewerten.

Die Auswertung zeigt, dass der Algorithmus zur Schätzung der Blicktiefe besser ist als einfache Raycasting und Strahlenschnittpunk-Algorithmen. Die objektive Bewertung der Anpassungsmethode zeigt jedoch, dass sie nicht dazu beiträgt, Müdigkeit zu reduzieren, sondern tatsächlich das Unwohlsein erhöht. Die subjektive Bewertung is nicht aussagekräftig, aber die Trends gehen in die gleiche Richtung.

Stichwörter Head-Mounted Displays, Vergenz, Augenermüdung, Eye-Tracking

Contents

1. Introduction

Stereoscopic 3D is a technique that creates depth impression in images by using binocular vision. The basic idea of stereoscopic displays is to display different images to each eye, faking the parallax that is usually produced by the offset of the eyes. Originally these displays produced a large amount of discomfort. This is partially attributed to binocular mismatches such as vertically offset cameras, depth inconsistencies (especially when doing 2D-to-3D conversions), and mismatches between different depth cues [1]. The main problem however is the well known vergence-accommodation-conflict [2]–[4]. This conflict arises when the eyes focus (or accommodation) remains on the screen while the eyes converge to the virtual distance of the object. As a mismatch between vergence and accommodation is unusual, it creates large amounts of discomfort. While most other problems have been solved, the vergence-accommodation-conflict remains unsolved. To work around this problem, most 3D-movies limit the range of object-depths to the so called convenience-band where the mismatch is small enough to not cause substantial eye strain [5]. This limits the overall depth perception.

More recently, head-mounted stereo displays, or HMD for short, have become more popular for both consumer and professional use. HMDs are worn on the user's head and use two screens close to the eyes to produce the two images to the eyes. Lenses between the display and the users eyes cause the eyes to focus at a fixed distance somewhere between $70cm$ and infinity depending on the HMD-model. This creates a immersive experience where the user feels as if he or she were in the virtual environment. As the eyes of the user still converge to the object distance, the vergence-accommodation-conflict remains present in HMDs and is the main hurdle when designing HMD applications.

Technical solutions have been proposed to adjust the accommodation by changing the optics between the screens and the eyes or physical movement of the screens [4], [6]–[9]. However, these systems are technically complicated and adaptable lenses have a limited aperture, limiting the field of view. Therefore they are not suited for the consumer market. A variety of rendering techniques have been shown to be ineffective [10]. Chapter 2 gives an overview of these methods.

A promising rendering technique is to adjust the object distance of the currently focused object to the accommodation distance, eliminating the mismatch between accommodation and vergence [11], [12]. However, while these studies did find a reduced discomfort, they did not determine the current focus of the participants. Instead, they assume the focus by either hand position or by telling the participants where to look. With the advent of eye tracking build into HMDs, solutions based on dynamic adjustment of rendering parameters based on the current vergence situation become viable. Therefore, this thesis examines the possibility to modify the interocular distance and vergence of the virtual cameras so that the vergence induced by a rendered object matches

the accommodation induced by the display. Chapter 4 gives details about this method.

The method requires an accurate measurement of the current eye convergence. Therefore, chapter 3 proposes a new method for this measurement that considers both the measured eye convergence and the geometry of the scene. The consistency of vergence and accommodation comes at the expense of dynamic modification of perceived absolute depth. Importantly, however, relative depth perception remains intact.

To evaluate the effectiveness of the approach, chapter 5 introduces an experiment based on a visual search task. The experiment evaluates the visual comfort relative to a baseline method that uses fixed interocular distance and parallel view directions for rendering. I am specifically interested in how matching of vergence and accommodation affects objective measures of discomfort such as pupil diameter and blink rate and subjective assessment based on self report. As part of this evaluation I also determine if participants notice the dynamic adjustment at all, and if so, how this behavior is judged. Chapter 6 details the results of the experiment. The data from the experiment is also used to evaluate the quality of the focus depth estimation.

2. Related Work

There is a significant amount of research about possibilities to solve or reduce the vergence-accommodation conflict. Traditional methods aim to minimize the conflict by remapping the disparity function of depth such that most scene content is viewed in a comfort zone [13]. More recently, methods that use information about the users gaze have been developed. The gaze can ether be measured using an eye-tracker or be estimated or assumed based on other factors. This chapter provides an overview over these methods. Table 2.1 summarizes evaluated aspects in each reference.

Changing the accommodation distance so that is matches up with the vergence is the most straight forward solution. As a 3D-scene naturally has different depths at once, accommodation should be different for different parts of the screen as well. Therefore, it is required to detect or predict the user's gaze. Physical movement of the stereoscopic 3D-screen based on the user's gaze location, as determined by an eye-tracker, has been proposed in 1996 by [4]. However, such a robotic system is large and expensive such that the authors did not manage to build a fully working system.

Changing accommodation is more feasible is an HMDs where virtual screen distance can be more easily changed using *adaptable lenses* (AL) [6]. These systems have been proven to improve depth perception [8] and visual comfort [10] and are preferred by users [8]. However, this solution suffers from a limited field-of-view because of limited apertures of the adaptable lenses used.

Monovision (MV) is a concept where each eye has a different lens and thus has to accommodate to different depths. These systems are cheap and are common in ophthalmology as a treatment for presbyopia (see [8] for a more in depth summary of its ophthalmological usage). Some studies have suggested that such a system could also be used for HMDs [8]. The authors found that monovision can increase visual clarity and depth perception. An advantage of this method is that it does not require information about the user's gaze. However other studies have found them to not increase comfort or depth perception in HMDs [10].

Rendering tricks can also be used to reduce discomfort produced by the vergence-accommodation conflict. [14] proposes, that *gaze-contingent depth-of-field* (DOF) could improve comfort by simulating the blurring of objects at different distances than the current vergence distance. They do find a statistically significant improvement of comfort but do recognize that users dislike such a blur. They speculate that this results from a slow update and the needed smoothing of the gaze location to reduce noise. Others find

Table 2.1.: An overview of the different aspects of the papers

	DOF	AL	MV	DC
user preference	[8], [14]	[6], [8]	[8]	[17]
depth perception	[8], [16]	[8]	[8]	[12], [18]
visual comfort	[10], [14]	[10]	[10]	-

either no significant benefit [8] or just a small benefit that might not justify the added computational load [10]. [15] extends on this by not only simulating depth-of-field but also chromatic aberiations that occur in the eye. They find a significant improvement over basic DOF-bluring. Gaze-contingent depth-of-field can also be used to enhance depth perception [16].

Changing the vergence in oppose to the accommodation (DC) is another possibility to solve the vergence-accommodation conflict. Changing vergence can be easily achieved by changing the disparity during or after rendering. Such a system has first been proposed by [11]. They propose to change the camera vergence to bring the current focal-object to zero disparity so that they are perceived at screen distance. An experiment to determine the speed at which the virtual camera convergence can be changed without the user noticing was conducted. However, their limited hardware and resulting flickering reduces the value of their findings.

Similarly, [12] uses camera convergence to the current virtual hand position (as an prediction of gaze-position). An experiment was conducted where users where asked to catch butterflies using a virtual hand with both static and dynamic vergence. The experiment showed that this convergence improves user performance. A questionnaire showed that none of the participants felt dizzy after either experiment and did not have a preference for either system. But the lack of a task to look for differences and a short experiment results in this subjective evaluation to be of limited value.

While the convergence of the virtual cameras is straight-forward for S3D displays, [17] deals with more complicated calculations for angled displays as typically found in HMDs. An informal experiment shows that convergence is preferred by users.

By using gaze data the disparity map can also be dynamically adjusted based on where the user is currently looking at. This has been demonstrated to improve depth perception [18].

3. Eye Tracking in Head-Mounted Displays

An eye-tracker is a device that measures the position of one or both eyes. Eye tackers have numerous applications. In fields like psychology, human factors and ergonomics, and marketing and advertisement eye trackers are often used in an diagnostic way. In this mode, the eye tracker only records the position/movement of the eyes for later analysis. In an active role the eye tracker results are used to interactively react to the users vision. Vision can be used as a pointing device or for adaptive (gaze-contingent) rendering [19, p. 205]. The application presented in chapter 4 is of the latter type while the experiment in chapter 5 also uses the eye tracker in an diagnostic way.

However, only few applications require an measurement of the eye convergence, i.e. the distance of the object currently focused. This chapter will describe how the current depth of the vision can be determined and presents an improved algorithm.

3.1. Eye Tracker Types

An eye tracker can measure the eye rotation either relative to the head or in space. Most applications require an absolute direction of vision - also called the "point of regard" [19, p. 51]. Different methodologies are used for these measurements.

Electro-OculoGraphy measures the eclectic potential of the skin around the eye. The potential-differences range from 15 to 200μV at $20^{\mu V}/_{deg}$ [19, p. 52].

Contact lenses provide a physical connection to the eye. Therefore this system provide a very sensitive measurement. Usually, a coil is embedded in the contact lens so that its movement in an electric field can be measured. However, the contact lenses need to be large to avoid slippage so that this method is rather intrusive [19, p. 53].

Video-OculoGraphy does not require contact with the eye or skin and is solely based on visual sensors. These systems use a camera to track the center of the pupil [19, p. 54]. In order to provide a point of regard measurement some eye trackers also track a corneal reflection of a fixed infra-red light source near the eye [19, p. 56]. More recently, eye trackers employed surface tracking to provide head movement measurements [20, p. 5]. Similarly, HMDs provide their own head-tracking so embedded eye-trackers only need to measure eye-in-head movement. In this theses, I will use the "HTC Vive Binocular Add-on" by Pupil-Labs [20].

Figure 3.1.: Calibration scene used to calibrate the eye tracker. Left: 2d calibration; Right: 3d calibration (left eye)

3.2. Calibration

In order to map the rotation of the eyes to a gaze direction in the virtual environment, the eye tracker needs to be calibrated [19, p. 70]. *Pupil labs* provides such calibrations for usage in HMDs as a unity package[1]. Two different types are available: 2D and 3D calibration. Figure 3.1 shows the calibration scenes. Both calibrations consist of one or more circles where targets are displayed at. The first target is in the middle of the circles. In 3D calibration multiple circles at different depths are used and the targets are shown at each depth in sequence before the next position is used. 2D calibration only uses one circle. When 3D calibration is used the eye tracker returns data in form of vectors that indicate the view direction of each eye. When 2D calibration is used the returned data consists of one 2D position in the view-space for each eye. A vector that describes the view direction can then be calculated based on the camera parameters. Changing the camera positions and rotations as proposed in this thesis would cause the vectors returned by the 3D calibration to be ambiguous. Therefore, I have decided to use 2D calibration.

The quality of the calibration can be determined by accuracy and precision. These are defined by *COGAIN eye tracker accuracy terms and definitions* [21] (as cited by [20]) as:

- "Accuracy [is] calculated as the average angular offset (distance) (in degrees of visual angle) between fixations locations and the corresponding locations of the fixation targets."
- "Precision [is] calculated as the Root Mean Square (RMS) of the angular distance (in degrees of visual angle) between successive samples to `xi, yi` to `xi+1,yi+1`."

[1] `https://github.com/pupil-labs/hmd-eyes`

Movement of the HMD on the head can decrease the accuracy of the calibration. Therefore it is important to recalibrate the eye-tracker when such a movement occurred. During a recalibration accuracy and precision can be calculated using the old calibration and the new data for verification of the old calibration. I have written a plugin for the pupil software to automatically calculate the verification accuracy and precision as well as the calibration accuracy and precision partially based on the accuracy visualization plugin. The code can be found on the accompanying DVD (see appendix B for details).

3.3. Baseline Algorithms for Gaze Estimation in 3D Scenes

In order to evaluate the performance of the new algorithm, first some baseline algorithms are needed. This section describes two simple algorithms: A simple raycast and a ray intersection algorithm. Both algorithms have advantages and disadvantages. My algorithm, as described in section 3.4, is build as a combination of these two simple algorithms, combining their advantages. Therefore, these two algorithms are a good baseline to compare to. All algorithms first map the eye tracking data to view-space for each eye using the eye-tracker calibration as described in section 3.2.

3.3.1. Simple Raycast

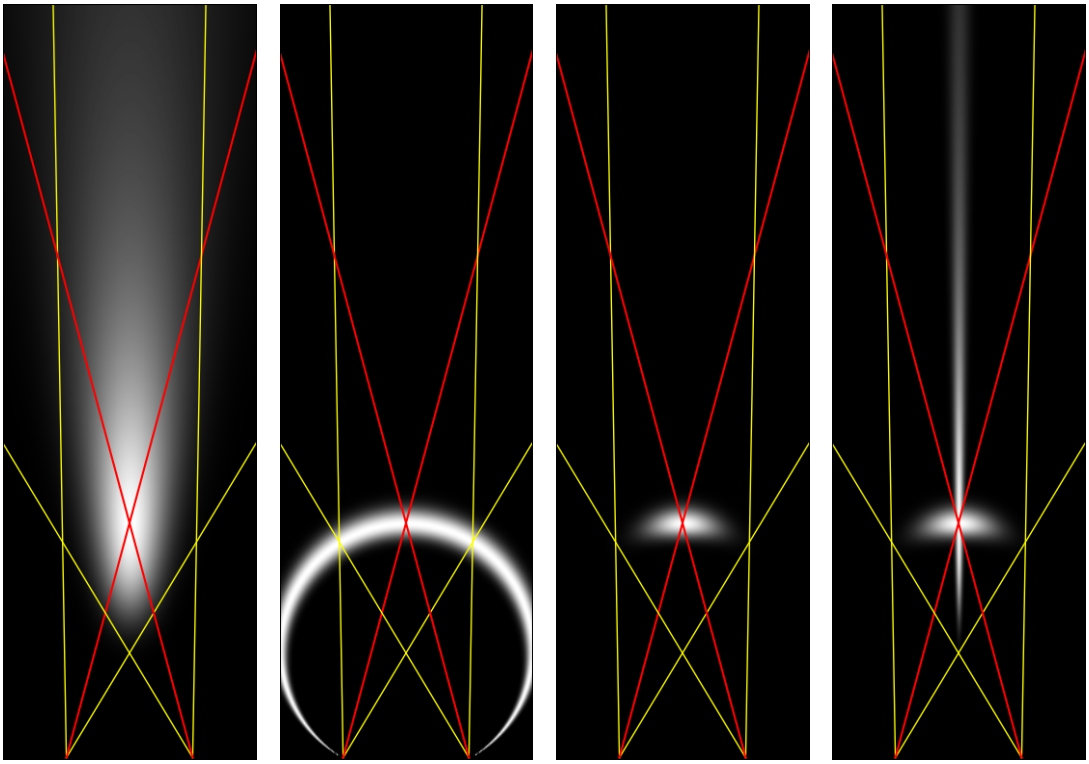
This algorithm uses a single raycast onto the geometry to determine the position the user is currently looking at. To determine the depth and position of vision a ray corresponding to one of the two eyes measured vision is cast. The intersection of the ray and the geometry of the scene is used as the current fixation and the distance to the camera is used as the fixation depth. However, the measured convergence of the eyes is not considered when doing this calculation. When the user is fixating at an edge or close to an edge, small error of the eye-tracker can lead to a large error of the calculated fixation depth because the algorithm wrongly assumes a fixation at the background.

3.3.2. Ray Intersection

In this algorithm, for each mapped eye-position a ray is created. Then the nearest points between these two rays are calculated. The center between these two points is used as the current fixation and the distance to the camera is used as the fixation depth [19, p. 77]/ However, the small distance of the eyes causes a large uncertainty of the fixation depth [22]. Because the geometry is not considered in the calculation, this can lead to the calculated point-of-regard not being on the surface of the scene.

3.4. Probability Based Gaze Estimation in 3D Scenes

This section presents an algorithm that uses both measured eye-convergence and scene geometry. The algorithm determines the position on the surface of the geometry that has the highest probability to be in focus. It is modeled based on a normal distribution



(a) independent (b) difference (c) combined (d) unsigned difference

Figure 3.2.: Probability of the eyes focusing on a specific point in space. Red lines are the measured direction; yellow lines represent 2 times standard deviation. (a) Product of independent normal distributions with standard deviation σ (b) Difference of error for each eye, modeled using normal distribution with standard deviation 0.15σ (c) Product of sub-figures a and b (d) Error that occurs when the direction of error is not considered in the difference

around the measured viewing direction of each eye. If the error measurement for each eye is considered independent then fig. 3.2a shows the resulting probability at each position in space. The red lines represent the measured view direction (mean) and the yellow lines represent a 2 times standard deviation error. The algorithm only considers points that are part of the geometry of the scene. Figure 3.2a also visualizes the large uncertainty in depth resulting from a relatively small uncertainty in direction.

However, a simple analysis of the recorded data from section 3.4.2 shows a high correlation between the X-errors of each eye (see table 3.1). It is therefore not possible to consider the eye-position measurements as two independent normal distributions. Instead, the difference of horizontal errors need to be considered. Figure 3.2b shows the probability for each position if only the difference between the X-axis errors is considered. Here the standard deviation of the normal distribution is smaller than in fig. 3.2a. A multiplication of these two distributions results in a reasonable estimate of probabil-

Table 3.1.: Correlation between X and Y axis errors for both eyes. Only the X axis errors of each eye are significantly correlated

	Left X	Left Y	Right X	Right Y
Left X	1.00	0.47	**0.86**	0.01
Left Y	0.47	1.00	0.20	0.20
Right X	**0.86**	0.20	1.00	0.24
Right Y	0.01	0.20	0.24	1.00

ities for each position as shown in fig. 3.2c. To avoid unreasonable asymmetry, both independent normal distributions for each eye are kept in the calculation.

The algorithm only considers points that are on the surface of the scene geometry and selects the point on the surface that has the highest probability, as described above, and is visible to both eyes. While it is possible that the user is looking at an object which is only visible to one eye, this scenario is relatively unlikely and an accurate fixation depth calculation is not possible in this case. Detecting and properly reacting to this condition might be an interesting further point of research.

3.4.1. Implementation

Because checking all points on the scene geometry is computationally too costly the algorithm uses raycasts from one eye position and then checks the visibility of the resulting point from the other eye. First, rays from one of the eyes (`dominant eye`) are cast in a square pattern around the measured direction. Both `search size` and `step size` of the pattern are configurable. For each ray, if the ray hit the geometry, the visibility of the hit-point from the other eye (off eye) is checked and the point is dismissed if it is not visible. If the ray did not hit anything, the background at infinite distance is visible in this direction. As the background is also a possible focal point, the other eye's visibility of the background in the same direction is checked and the direction is dismissed if the background is not visible. Finally, the probability for each remaining point is calculated as

$$P = norm(\alpha_{left}) * norm(\alpha_{right}) * norm_{diff}(\alpha^*_{left} - \alpha^*_{right}) \tag{3.1}$$

where α_{left} and α_{right} is the angle between the ray from the left or right eye respectively to the point and the measured direction. $norm(x)$ and $norm_{diff}(x)$ are different normal distribution functions with mean 0 and a configurable standard deviations. Angles in 3D-space are usually always positive but signed angles are necessary to calculate the difference probability in order to avoid the problem highlighted in figure 3.2d. Therefore, α^*_{left} and α^*_{right} are signed angles in the horizontal axis.

The implementation in Unity is based on a custom 3D-camera rig as described in appendix A. To improve performance, the calculation is done on multiple threads using the unity job system. The job pipeline consists of five jobs. The first job creates the rays for the dominant eye, the second job is the unity raycast execution, the third job uses the raycast hits and prepares the raycasts to determine visibility by the off eye, the

Table 3.2.: Parameters of the eye-tracker algorithm

Parameter	Range	Explanation
`dominant eye`	left, right	eye to start calculation at
`search size`	$\mathbb{R}^+$	size of square around view direction to consider
`step size`	[0..`search size`]	distance between rays to consider
`sigma`	$\mathbb{R}^+$	standard deviation of normal distribution
`sigma_diff`	$\mathbb{R}^+$	standard deviation used for difference

fourth job casts these rays and the fifth job uses the result of both raycasts, calculates the probability and determines the point with maximum probability. The complete code can be found on the accompanying DVD (see appendix B for details).

3.4.2. Optimization

As the algorithm has a number of configurable parameters, these need to be optimized. The parameters are summarized in table 3.2. For search size and step size there needs to be a compromise between accuracy and performance. The dominant eye setting does only matter if the step size is large. The standard deviations are both only used in calculation of the probability of a point in Formula 3.1. Simplification yields

$$P = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{{\alpha_{left}}^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{{\alpha_{right}}^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi\sigma^2_{diff}}} e^{-\frac{(\alpha^*_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}} \tag{3.2}$$

$$= \frac{1}{(2\pi)^{\frac{3}{2}}\sigma^2\sigma_{diff}} e^{-\frac{{\alpha_{left}}^2}{2\sigma^2}} \cdot e^{-\frac{{\alpha_{right}}^2}{2\sigma^2}} \cdot e^{-\frac{(\alpha^*_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}} \tag{3.3}$$

$$= \frac{1}{(2\pi)^{\frac{3}{2}}\sigma^2\sigma_{diff}} e^{-\frac{{\alpha_{left}}^2}{2\sigma^2}-\frac{{\alpha_{right}}^2}{2\sigma^2}-\frac{(\alpha^*_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}}. \tag{3.4}$$

The algorithm determines the ray direction (α_{left} and α_{right}) where P is maximized. Because σ and σ_{diff} are constant, the maximum of P is at the same position as the maximum of the power of e in the above formula.

$$argmax(P) = argmax\left(-\frac{{\alpha_{left}}^2}{2\sigma^2} - \frac{{\alpha_{right}}^2}{2\sigma^2} - \frac{(\alpha^*_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}\right) \tag{3.5}$$

$$= argmin\left(\frac{{\alpha_{left}}^2}{2\sigma^2} + \frac{{\alpha_{right}}^2}{2\sigma^2} + \frac{(\alpha_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}\right) \tag{3.6}$$

$$= argmin\left(\frac{{\alpha_{left}}^2+{\alpha_{right}}^2}{2\sigma^2} + \frac{(\alpha^*_{left}-\alpha^*_{right})^2}{2\sigma^2_{diff}}\right) \tag{3.7}$$

Let $f := \frac{\sigma_{diff}}{\sigma}$.

$$argmax(P) = argmin\left(\frac{{\alpha_{left}}^2 + {\alpha_{right}}^2}{2\sigma^2} + \frac{(\alpha^*_{left} - \alpha^*_{right})^2}{2f^2\sigma^2}\right) \tag{3.8}$$

$$= argmin\left(\frac{f^2({\alpha_{left}}^2 + {\alpha_{right}}^2) + (\alpha^*_{left} - \alpha^*_{right})^2}{2f^2\sigma^2}\right) \tag{3.9}$$

$$= argmin(f^2({\alpha_{left}}^2 + {\alpha_{right}}^2) + (\alpha^*_{left} - \alpha^*_{right})^2) \tag{3.10}$$

This shows, that the output of the algorithm is not dependent on σ and σ_{diff} separately but only the variable f, so only f needs to be optimized. This optimization is done by simply testing the system with different values and determining the minimum of the average error.

3.4.3. Execution of Optimization

To optimize f, I have created a simple test scene as shown in figure 3.3. I have then recorded the eye-position as reported by the eye-tracker so that every configuration can be tested against the same data. The test is a static 3D scene with a number of different targets that the user is looking at in order to record the data. During the recording any head-tracking is disabled so that the user is focusing on the targets using eye-movement and not head-movement. The scene consists of a central pillar that creates a large depth difference to the background and a larger block with only a small depth difference to the background plane. There is also a column though the whole scene from top to bottom on the left of the the central pillar that is used to test the algorithms ability to detect focus on small foreground objects. The test targets are located as follows (see figure 3.3): 9 targets on the central pillar (one in the center, one on each edge and one on each corner), 8 Targets around the central pillar visible to both eyes (one on each edge and one on each corner), 3 Targets on the column, 3 Targets on the background next to the column and 3 on each side of the central pillar visible to only one eye each.

Recording of the test-data was done using a HTC-Vive Pro HMD and the aforementioned eye-tracker add-on by *Pupil labs*. Before the recording is started, the eye-tracker is calibrated using the default calibration settings in 2D mode. The recording starts with the first target being marked using a red light, keeping the texture of the object visible. When the user is ready and looking at the target, he or she is pressing the space bar for one second. During this time the eye movement is recorded and the target is changing color to indicate the recording process. If the user is not comfortable with the recording, e.g. when they are accidentally looking away, they can let go of the space bar cancelling the recording. If the recording is successful, the next target is selected. The user has to let go of the space bar and then repress it when he or shw is ready to record the next target. The recorded data is saved as a simple `csv`-file containing target-name and x and y position for each eye. The recorded data can be found on the accompanying DVD (see appendix B for details).

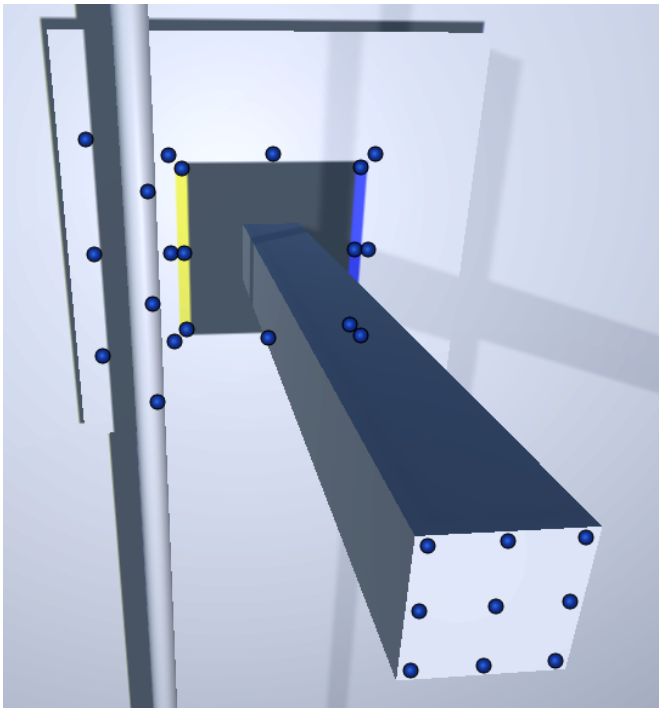


Figure 3.3.: Test Scene used to test the quality of the eye-tracking depth calculations. The camera is located in front of the large pillar in the middle. The blue dots represent the reference locations. The yellow and blue areas are only visible to the left or right eye respectively.

Evaluation is done by comparing the output of the algorithm with the position of the target. The recorded eye-tracker data is read and the pupil-framework is faked to give the recorded data instead of the true eye-tracker data. The system then waits for the algorithm to calculated the new result and calculates positional and depth error. These errors are then averaged over all recorded positions and targets.

Results: The measurements for different f-values can be seen in fig. 3.4. An additional measurement only considering targets visible to both eyes is also shown in the figure. A quick reduction of the average error for f-values below 0.14 can be observed. Beyond $f = 0.16$ the average error slowly increases again if all targets are considered. When only the target visible to both eyes are considered, the average error only increases again for $f > 0.18$. Because of the cliff-edge at $f = 0.14$, and in order to reduce the risk of slight errors in this measurement to lead to a significantly increased average error, I use $f = 0.15$ in the following chapters.

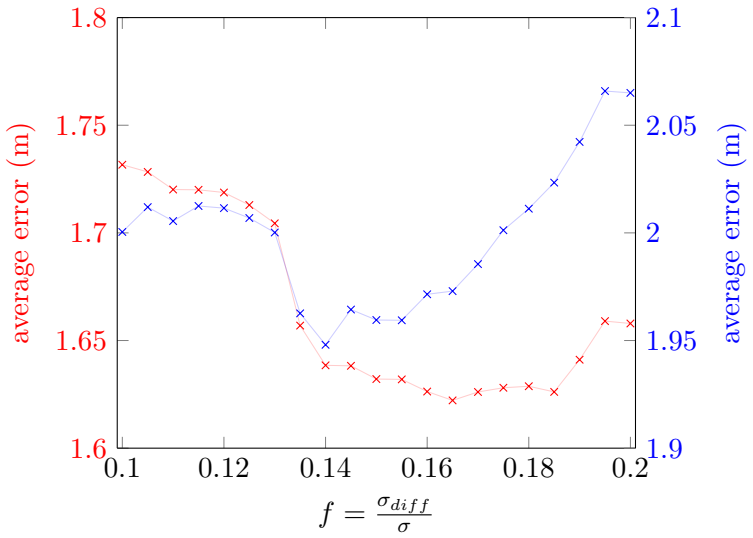


Figure 3.4.: Average error for different f values. only considering targets visible to both eyes; considering all targets

3.5. Conclusion

In this chapter I have presented the basic functions of eye-trackers. As a good depth estimation from eye-tracker data is difficult and no good method exists, I have proposed a new algorithm that combines the benefits of simple raycasting and ray intersection calculation. I have then optimized the algorithm using a simple test scene. A comparison of the improved algorithm to simple raycasting and ray intersection algorithms is done in chapter 6 based on data from the experiment presented in chapter 5.

4. Adaptable Stereoscopy

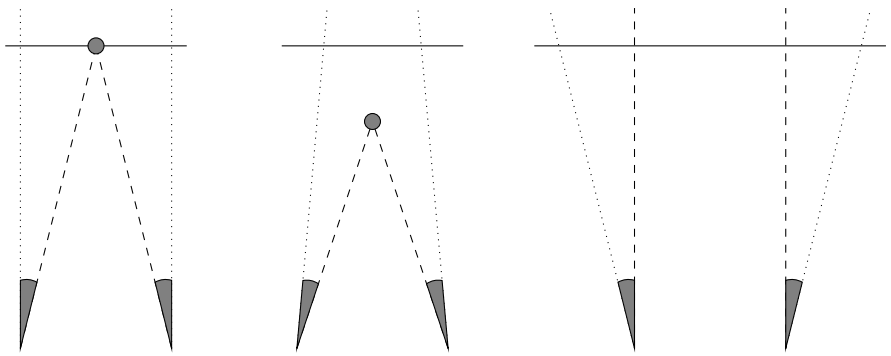
Most traditional stereoscopic media uses depth maps to keep everything inside the convenience band. The main idea presented in this thesis is to adapt the stereoscopic rendering in accordance to the currently focused object by the user. This chapter lies out two different possibilities of how this can be achieved. Subsequently, the proposed protocol for combination is described, followed by a discussion of different methods to change the camera parameters. Finally, the maximum change speed for continuous change is discussed.

4.1. Background in HMD Rendering

Rendering to a HMD is different than rendering to a standard stereoscopic monitor as the screens displaying the images for each eye are located at slightly different positions exactly in front of each eye. As such, objects that are rendered at the same display coordinate are perceived at a distance of infinity, whereas they are perceived at the screen distance on stereoscopic monitors. On an HMD objects which are located at the ideal distance (the virtual screen distance) are displayed at different coordinates. Thus, the ideal eyes convergence is different than the virtual cameras convergence. Usually, the virtual cameras for HMD rendering are parallel, rendering objects at the distance that they actually are.

In the following considerations, the convergence depth refers to the distance where the eye convergence is optimal. In order to calculate the actual camera rotation a correction angle θ has to be applied. This angle is calculated as the angle between the forward direction and center point on the virtual screen distance. A separation of $0.069m$ and a virtual screen of $2m$ yields a correction angle of $\theta = 0.988°$. Figure 4.1 shows (a) this calculation, (b) how his angle can be applied to calculate the true camera location when a specific eye convergence is intended and (c) the maximal divergence of the cameras needed such that objects at infinite distance can be viewed with optimal eye convergence.

On stereoscopic monitors, the distance between the coordinates a object is displayed at for each eye is called *disparity* [23]. So objects rendered at the screen distance have a disparity of 0 while objects closer to the screen have a negative disparity and objects farther away have a positive disparity. In order to extend this definition for HMDs, the *disparity* in degrees shall be the difference between the signed angles between the camera forward directions and the object location offset by $-2 \cdot \theta$ such that objects at virtual-screen distance in front of the camera have a disparity of $0°$.



(a) default position (b) close convergence (c) infinite convergence

Figure 4.1.: On HMD displays the eye do not converge to the same point as the virtual cameras converge. This can be corrected with the correction angle θ. (a) The default position used to calculate θ; (b) application of correction angle for close convergence; (c) application of correction angle for convergence at infinity yields maximal divergence of cameras of $2 \cdot \theta$

4.2. Adaptation Methods

In order to reduce the effect of the vergence-accommodation-conflict, the currently focused object should be rendered as if it were at the same distance as the virtual screen distance as eye accommodate to this distance. Alternatively, objects can be rendered slightly in front or behind that distance as long as this distance is inside the convenience band. Apart from dynamic disparity map adjustment as done by [18], there are two straight forward methods to achieve a correct eye vergence: changing the camera separation and changing the camera vergence. This section describes these two possibilities. For the eye-tracking algorithm described in chapter 3 to work, the user needs to be able to look at all objects at all times. This is limited by the amount of eye-convergence possible in the HMD. Using the largest lens to eye distance, as required for the eye-tracker, the HTC Vive has a field of view for each eye of 39° and 37° respectively[1] resulting in a theoretical maximum convergence of 38°. A simple test has shown a maximal convergence of about 21°. For convenience divergence of the eyes should never be needed. Indeed, a simple test shows that the brain sometimes can not fuse the images when eye divergence is needed. In other words, every object should always have a disparity between $-2 \cdot \theta$ and 21°. The black line in fig. 4.2 shows the disparity for each distance using the default rendering.

4.2.1. Changing Convergence

For this option, the cameras can be converged to the desired depth as shown in figs. 4.3a and 4.3b. As this is the natural movement of the eyes for looking at close and far objects

[1] see `http://doc-ok.org/?p=1414`

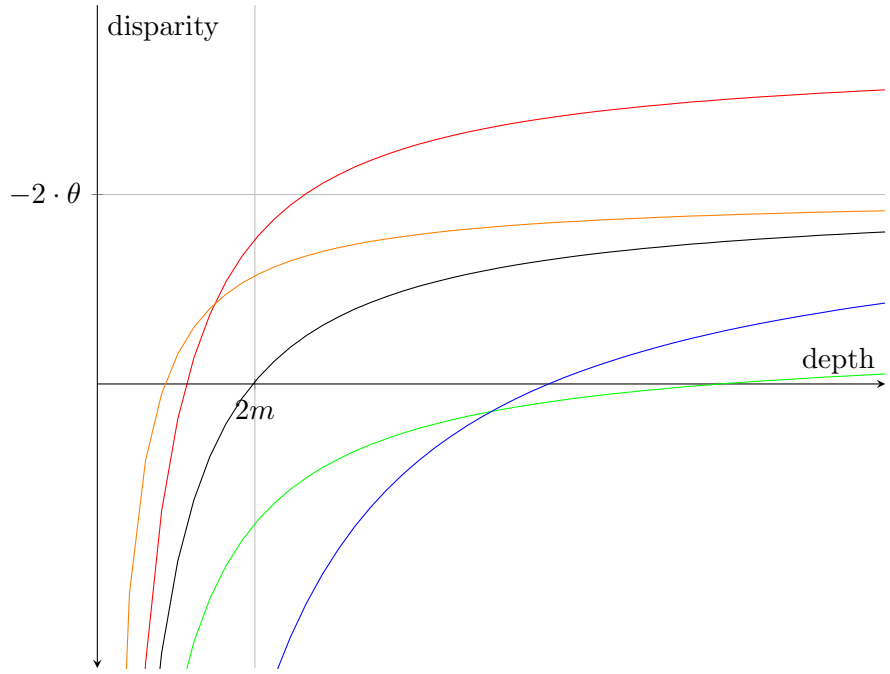
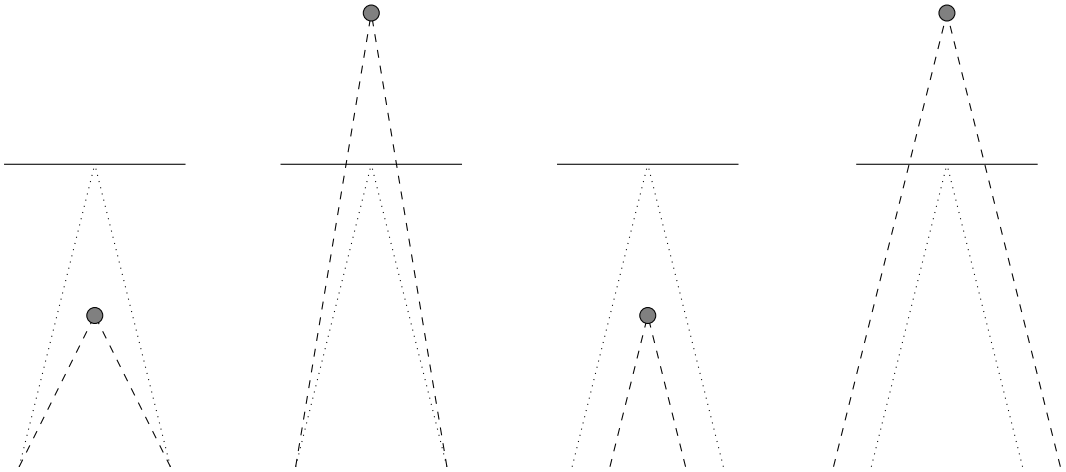


Figure 4.2.: The disparity function of depth. The black line shows the default disparity map. The orange line draws the mapping when distances between two cameras are small while the blue line depicts the mapping when camera distances are large. The green line corresponds to diverging cameras while the red line corresponds to converging cameras.

this adaptation does not change the fundamental perception of the scene. Instead of the eyes converging the eyes can stay at the optimal convergence (at virtual screen distance) while the virtual cameras converge instead. As shown by the red and the green line in fig. 4.2 this method adds or subtracts the same amount of disparity for all objects. Although this option is the most straightforward option there are a few problems with this approach:

If the stereo cameras are not parallel, the scene is projected onto different planes. This leads to keystone distortions [24] where the same object is drawn at different sizes for each eye, leading to visual discomfort [18]. As a solution [1] recommends keeping the projection plane stable and only moving the part of the plane that is drawn. The authors also found that these keystone distortions do not have a large effect on visual comfort. Therefore, I have not followed this recommendation in this thesis.

The second problem occurs only when converging to points closer than the virtual screen distance as in fig. 4.3a. In this case, the actual cameras rotation converges as shown in fig. 4.1b. If the user were then looking at an object at a distance farther away than the real camera's convergence depth, his or her eyes would need to diverge as such an object is rendered to the left of the center for the left eye and to the right of the center for the right eye.



(a) convergence close point (b) convergence far point (c) separation close point (d) separation far point

Figure 4.3.: Possible adjustment methods by camera adjustments. The lines represent the eye direction that needs to be adjusted by θ to determine the actual camera rotation. Dotted lines are default position, dashed lines represent the adapted view.

When adapting for vision at infinity, the cameras diverge to a maximum of $2 \cdot \theta$ as shown in fig. 4.1c. Therefore, convergence of the eyes is only increased marginally such that it is still possible to focus at all close objects.

4.2.2. Changing Separation

The second option is to move the stereo cameras on the base line, changing their separation as shown in figs. 4.3c and 4.3d. Changing the distance of the eyes is not realistic but can improve depth perception as the parallax increases [25]. Here, the camera separation is changed so that the cameras converges at the correct distance while not changing the rotation. As shown by the orange and the blue line in fig. 4.2 the disparity for far away objects is changes less than for close objects. This method also does not suffer from keystone distortions and solves the eye divergence problem for close objects. However, for very far objects the camera needs to be very far apart resulting in the convergence angle, needed for very close objects, being too large, as illustrated by the blue line. As a result, it is impossible to focus on close objects. For objects at infinite distance this option does not work as it would require infinite separation.

4.3. Combination of Adaptation Methods

The two methods can be combined by adapting camera convergence and separation at the same time. As both methods manipulate the cameras to converge at the correct depth,

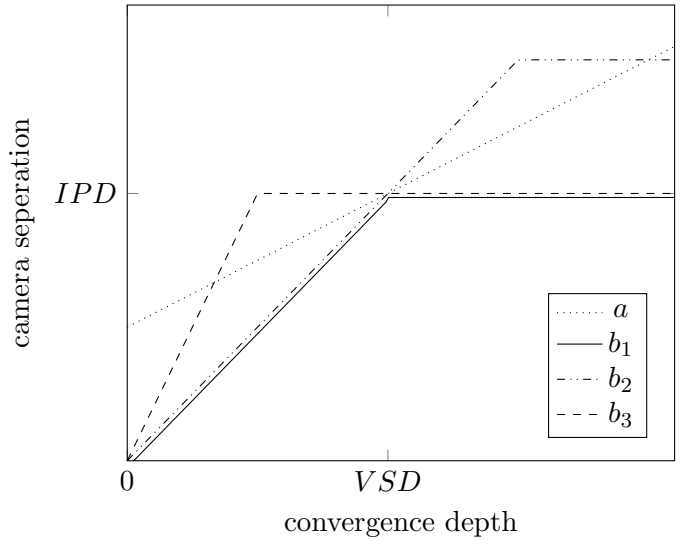


Figure 4.4.: Possible functions describing the camera separation over convergence depth: a using a constant interpolation; b_1 default using separation before IPD and convergence after; b_2 alternative when depth perception is important; b_3 alternative for scenes without very far objects

the only difference between the two methods is the camera separation. When changing convergence, the camera separation remains constant and the cameras are rotated to converge correctly. With the changing separation method, the camera separation is changed in a way that the cameras don't need to be rotated for the right convergence. In order to combine the two methods, I simply choose some separation between the default eye-separation and the separation required for the changing separation method and rotating the cameras to converge correctly. Keeping the fraction of the way between these two points constant, as shown in function a in fig. 4.4, results a change in both separation and camera rotation. This can combine the benefits of both methods. At far distances the cameras do not separate as much while still providing an increased parallax for 3D vision. At close distance the actual camera convergence is reduced, limiting the amount of possible eye divergence. However, all possible problems of both methods still exist: Objects at infinity would still result in infinite camera distance, and changing vision from a close object to a very far object still results in some eye-divergence.

In order to solve these problems, I use a more adaptive approach of choosing the camera distance. Figure 4.4 shows possible functions that define the camera distance depending on the object distance. Function b_1 shows the simplest solution: The camera distance is proportional to the object distance up to an object distance equal to the virtual screen distance. At this point the camera distance is equal to the default camera distance. For object distances larger than the virtual screen distance the camera distance remains constant. Thus, this solution uses exclusively the changing separation method for objects closer than the VSD and exclusively the changing convergence method for

objects farther than the VSD, i.e figs. 4.3b and 4.3c. Most of the problems described above are solved using this method. The actual camera rotation never converges so that eye divergence does not occur and very far objects do not cause a very large camera separation. Depending on the application, other functions might be suited: A camera distance multiple times IPD doesn't pose a problem if the scene does not include very close objects, so the camera separation might be increased beyond the IPD to increase depth perception. The graph b_2 in fig. 4.4 shows such a function. If the scene does not include very far objects and only a single color or no background, some actual camera convergence does not result in eye divergence. In this case, keeping the camera-separation larger could result in improved depth perception so a function like b_3 in fig. 4.4 might be preferred.

4.4. Implementation of Dynamic Changes

Simply knowing the optimal camera configuration is not sufficient as there needs to be some transition between the current and calculated optimal camera configuration. There are multiple options for this transition. This section explores these options and details there benefits and drawbacks:

Instantaneous change of the camera separation and convergence to the optimal position reduces the VAC as much as possible. However, this causes the camera movement to not be continuous. Discontinuous movement can cause problems when the user is tracking objects and cause confusion and disorientation. In contrast [26] found that with fast enough response a change is not noticeable.

Blinking cause temporary brakes in vision that can be used to hide instantaneous changes. Blinking can be very easily detected by an eye-tracker and occurs in regular intervals. The optimal camera configuration can be calculated before the blink and only be updated once the blink occurs. However, this causes some time between the saccade and the blink, where the camera parameters are sub-optimal. Also, my eye tracker system was not able to detect blinks fast enough.

Saccades themselves can also be used to hide the instantaneous change [27]. This method is very time sensitive as saccades only last a few milliseconds in which the adjustment needs to occur. As a saccade changes the focus depth of the user, it is not possible to determine the optimal camera parameters beforehand. This creates a very small time window to calculate and change the camera. Also, while methods predicting the saccade destination during the saccade exist [i.e. 28], the accuracy of these predictions is inferior to the position determination after the saccade [18].

Continuous change does solve the confusion problems of the instantaneous change. Depending on the speed of the change this can lead to a large amount of time in which

the camera convergence is not optimal. A faster change would be noticeable and would lead to confusion and disorientation. The following section will investigate the optimal change speed.

Because of the difficulties of the usage of saccades and blinking I only consider continuous change in this thesis.

4.5. Change speed

Changing the camera parameters might not be necessary if the focused distance only changes slightly and still is in the convenience range. For example, A change can only be applied if the focused distance is outside the convenience band. Then the camera parameters could be adjusted to either the correct distance or just enough to put the focused distance in the convenience band again. The first method would create inhomogeneous changes and the second method would create inconsistent camera parameters when looking at the same object. A simple test has shown that these problems make the adjustment more noticeable. Therefore, I have decided to always change the camera parameters.

Continuous change can happen at different speeds. In order to avoid confusion the goal is to avoid the change to be easily noticeable by a naive user. [18] found that camera convergence of $17.64^{arcmin}/_{s}$ is the median just not noticeable speed. As the adaptation presented here also includes camera separation changes, I have decided to use a fixed change of the inverse of the camera convergence depth of

$$c_0 = 0.2\frac{1}{m \cdot s}. \tag{4.1}$$

A simple experiment has shown this speed to not be noticeable while being fast enough to catch up to the users vision.

In every frame the camera convergence distance is updated to

$$D_{new} = \frac{1}{\frac{1}{D_{old}} \pm c_0 \cdot t}. \tag{4.2}$$

where t is the time passed since the last frame. The new distance is then used to calculate the camera parameters as described in section 4.3.

4.6. Conclusion

In this chapter I have detailed the possibilities of camera settings to affect the depth of the currently looked at object. I have found that camera separation should be used for close objects and that camera vergence should be used for far objects. Furthermore, I have explored different possibilities of changing the camera settings to achieve the best settings. Fillally, only a slow continuous change is feasible with my setup. The next chapter will present an experiment to evaluate the visual comfort of participants using this method. Chapter 6 will present the results found during this experiment.

5. Experiment

The evaluation of the adjustment method is done using a experiment. The goal of the experiment is to evaluate if the tiredness or fatigue of the participants is changed when the adjustment from chapter 4 is applied. Furthermore, the data of the experiment should be able to be used to evaluate the quality of the eye tracking depth calculation from chapter 3. This chapter outlines the design of the experiment and details its execution.

The experiment is based on the Dobble game. The *Dobble* game[1] (also known as *Spot it!*) is a card game consisting of cards with usually eight items displayed each. The cards are designed in such a way that for each pair of cards there is always one common item. The task of the players is to identify the common item of the two cards presented.

The benefit of using this design is that the user concentration can be measured: Because the user needs to look at each cards it is possible to detect how often the participants needs to switch the vision between the cards. A simple visual identification task does not provide such a measurement.

For the experiment, each card (in the following also referred to as disk or target) displays 5 symbols from a list of selected Unicode symbols from the *Miscellaneous Symbols* block ranging from U+2600 to U+26FF. Figure 5.1 shows an example disk pair.

5.1. Test Scene

The test scene consists of an empty room with a table and a window behind the table as shown in fig. 5.2. The dimensions of the table in the scene is set to match the physical dimensions of the table used in the lab. This is to allow the participants to physically feel the table while wearing the HMD, minimizing confusion.

[1] `https://asmodee.de/dobble/dobble`; accessed Jun. 23, 2019 10:25; german

Figure 5.1.: Example of a Dobble game. The common symbol needs to be selected.

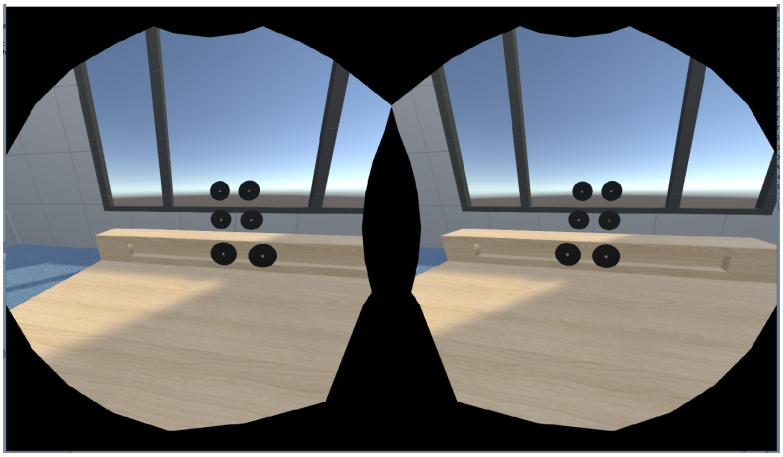


Figure 5.2.: The scene used for the experiment. Only targets at one depth are shown at once.

For each round, the two disks are shown at one of 3 depths. A close depths at a ledge of the table at about $0.46m$, a middle one at the window at $2m$ and a far depth at $50m$ visible through the window. Because the cameras are moved around to follow the participants movement and the participant location is not fixed these distances can be slightly different and change during the experiment. The size of the disks are dependent on the distance they are shown at. They are sized to the same apparent size: All targets have a visual angle of about 8.3°.

5.2. Gameplay

Once the correct symbol is identified, the participants can select it by using the touchpads on the HTC Vive controllers. Each controller is used to move a visible cross on the targets to the correct symbol. The symbol, where the cross is currently at, is slightly enlarged to visually indicate its selection. Once both crosses are at the correct location the participants can press down on the touchpads to select the symbols. If the selection was wrong (i.e. two different symbols were selected), the color disks is briefly changed to red to provide visual feedback to the participants. In either case, the symbols are then hidden. The participants then need to move their thumbs (and thus crosses) back to the middle of the disks for the next symbols to appear. This is indicated by a small circle in the middle of the disks. Moving the thumbs back to the middle is necessary to prevent correlations between the correct symbol location of adjacent rounds to affect the solution time. The controllers themselves are not visible while playing the game to avoid the participants to looking them, as this would cause additional unwanted camera adjustments. After every five sets of disks the distance is switched.

5.3. Protocol

Because the performance of different participants varies significantly, it is important to allow within-participant comparisons. Therefore, every participant did two sessions: One with and one without the adjustment method enabled. I expected that participants learn and get better at the game over time. To balance out this effect, one half of the participants had the session with adjustment first and the other half the session without the adjustment. All participants were shown identical disks in the same order. Also, both sessions were identical apart from whether the adjustment was enabled or not.

Each session consists of 6 blocks. Between each block as well, as before the first block and after the last block, the eye tracker calibration was run. The calibration data was also used to validate the last calibration as described in section 3.2. Each block consists of 35 individual disk pairs shown (below referred to as trial), grouped into groups of 5 that are shown at the same depth before the depth was changed. A block is 3 minutes and 30 seconds long. If the participant completes the 35 trials faster, he or she will need to wait for the remaining time to pass. If he or she is slower, the remaining trials will be skipped. In total, each session is about 20 minutes long. The order of the distances the groups are shown at is laid out such that every depth change (close to middle, close to far, middle to close, middle to far, far to close and far to middle) occurred exactly once. In order to have every jump represented by every participant - even the slower ones whose last trials of each block might be skipped - the order of the depth changes is altered between blocks. The complete list of trials with their corresponding depth can be found in the database on the accompanying DVD (see section 5.4 and appendix B for details). As encouragement, a score is displayed at the end of each block that is calculated as the sum of the points of each trial. Each wrong answer is penalized by -100 points, while every correct answer is rewarded $1000/(t+10)$ points, where t is the time needed for the trial in seconds. Participants achieved scores between 1151 and 2980 points (median 2587 points) in each block.

Before the experiment, the participants were asked to fill in their personal information in an online form. Each participant spend a total of one hour for the experiment. After giving their written consent, they were first shown how the calibration works by doing a practice calibration. This practice calibration was also used to change the calibration circle to be centered in the participants view. Any other problems with the calibration were addressed and fixed before proceeding further. Subsequently, the participants were given a chance to get familiar with the game and the touchpad controls. This was not done in the HMD but on the screen to avoid any additional fatigue. Once the participants felt comfortable with the controls, the first session was started. After the first session, the participants were asked to complete a questionnaire, as described in section 5.6, followed by a ten minute break. After the break the second session was done, followed by another questionnaire. In the end, the participants received their 10 € compensation after completing the necessary paperwork. For some participants, an additional FLY stereo acuity test (Vision Assessment Corporation) was done.

Figure 5.3.: Database schema used to save experiment design and recording. Fields marked with (*) include multiple fields in the real database that where omitted here for simplicity. Recorded values can be found in section 5.5; Questionnaire questions can be found in section 5.6.

5.4. Database

I employ a database implemented using SQLite to store the trials in the experiment as well as information about the participants and the measurements as described in the following two sections. This is done to simplify the data analysis. The complete database schema is shown in fig. 5.3. All trials are saved in the `dobble_round` table, are associated with a preset, and are ordered using the `nr` field. The `preset` table is used to store all used preset. This experiment uses the same preset for all sessions but a different preset is used for the example run as descried above. Five trials are grouped into one `distance_group`. The `distance_group` table stores the distance the group is displayed, the block it is in, and the jump used from the last block to the current one.

Each participant is entered into the `participant` table. Their answers in the questionnaires are entered into the `questionnaire_responses` table (see section 5.6). Both sessions are individually entered into the `session` table and given a preset to use as well as their method (with or without adjustment enabled) and the number of the session for the participant (1st or 2nd). The `dobble_round_solution`, `blink` and `recording` tables are used to store the objective measurements (see section 5.5).

5.5. Objective Measurements

For an objective evaluation, a number of parameters were recorded during the experiment. Not all recorded parameters were used for the analysis.

Blinking is recorded in the `blink` table, where every blink is saved with a timestamp and a confidence value.

Furthermore, there are parameters that are recorded once for each trial and parameters that are saved periodically 60 times per second.

5.5.1. Per Trial Recorded

Per trial recorded parameters only need to be saved once for each trial. They are saved in the `dobble_round_solution` table in the database.

Timestamps of the start and end-time in milliseconds since the start of the experiment. This data can later be used to calculate the time needed for completion.

Selected symbol for left and right disk as entered by the participant. The answer can later be compared to the correct solution to determine whether the participant has made an error or not.

Points scored in the trial as described in section 5.3.

5.5.2. Periodically Recorded

Peroodically recorded parameters are recorded 60 times per second into the `recording` table. In this table, each recording is associated with a `dobble_round_solution`.

Camera position and rotation as reported by the HTC-Vive tracking. This data can later be used to reconstruct the cameras, and evaluate the head movement.

Camera distance and convergence are also needed to reconstruct the camera positions and can also be used to verify the algorithm.

Eye position for each eye in view-space coordinates. This information is useful to measure eye movement and the eye movement speed.

View rays for each eye as origin and direction vectors. While view rays can be calculated from camera position and eye position, having this data directly available outside of the game simplifies the analysis of view convergence.

Gaze point as calculated by the algorithm presented in section 3.4. Useful to verify the algorithm output.

Selection cross positions for each disk might be used to measure indecisiveness of participants.

5.6. Questionnaire / Subjective Measurements

For a subjective evaluation of the participants fatigue and eye stress, two questionnaires are used. The first questionnaire is filled out after the first session and the second questionnaire is filled out after the second session. All questions from the first questionnaire are included in the second questionnaire but the second questionnaire includes additional questions about a comparison of the two sessions.

5.6.1. Questions about Individual Sessions

These questions (block A) are asked in both questionnaires about the respective preceding session. The participants are asked to evaluate them on a 5 grade scale from 1, representing very good, to 5, representing very bad. These questions are:

(A1) How tired are your eyes?

(A2) How clear is your vision?

(A3) Is it easy for you to focus now?

(A4) How does your head feel?

(A5) Do you feel comfortable?

(A6) How do you like this experiment session?

(A7) How difficult is the task?

(A8) Did the task get easier over time?

Questions A1 to A5 aim at evaluating the participants fatigue after the session. Questions A6 aims at evaluating the participants motivation and/or frustration with the game. Questions A7 and A8 aim at evaluating self-perceived performance. On the 5 grade scale of the latter question, 3 represents no change, while 1 represents the task getting easier at the end and 5 represents the task getting more difficult at the end.

5.6.2. Questions Comparing Sessions

These questions (block B) only appear on the second questionnaire and aim at evaluating subjective differences noticed between the two sessions. First, the participants are asked whether or not they have noticed any differences between the sessions and if so, to write down what differences they notices.

Then the participants are asked to answer a number of specific questions comparing their fatigue between the two sessions. Here, again, a 5 scale grade is used, where 1 means Session 1, 5 means Session 2 and 3 represents no difference. These questions are:

(B1) Which session was most fatiguing?

(B2) Which session irritated your eyes the most?

(B3) If you felt headache, which session was worse?

(B4) Which session did you prefer?

(B5) Which session is easier to play?

(B6) If you have to choose to repeat one session, which one will you choose?

(B7) Which experiment session is easier to change between different depths?

(B8) Which experiment session has more realistic depth impression?

Questions B1 to B6 are re-asking similar questions as block A, but might produce less noise as participants can directly compare the two sessions. Questions B7 and B8 are only asked after the second session to avoid the participants trying to identify depth changes during the second session. Furthermore, the participants are asked whether they noticed any camera movement during one of the sessions, and if so to name what movement they noticed.

5.7. Participants

A total of eighteen participants took part in the study for a compensation of 10€. Exept three of them, who have agreed to participate without the compensation, as they could not provide the necessary information needed to receive the compensation. The participants were kept naive as to the purpose of the experiment. Seventeen participants were students of the TU-Berlin. Five participants were female. The mean age was 24 with a standard deviation of 3.7. Participants could not be wearing glasses because glasses could negatively affect the eye tracker as well as concerns about physical space inside the HMD. Three participants wore contact lenses, all other participants did not need visual correction for HMD usage. Fourteen participants reported they had previous experience in VR. Their subjective evaluation of their prior experience with VR on a range between 1 (very bad) and 5 (very good) was 3.8.

5.8. Conclusion

This chapter presented an experiment that aims at evaluating the visual fatigue of the participants using the adjustment method presented in chapter 4. Both subjective and objective measurements are taken. In the next chapter, the data recorded will be analyzed and the results will be presented.

6. Analysis and Results

This chapter outlines my analysis of the data recorded in the preceding chapter and presents the results of the analysis. First, the performance of the proposed eye tracker gaze depth estimation is evaluated. Second, the objective measurements are evaluated, followed by the questionnaire answers. Finally, possible improvements are discussed.

6.1. Evaluation of Gaze Estimation Accuracy

To evaluate the quality of the proposed probability based gaze estimation from section 3.4, the method is compared to the two baseline algorithms described in section 3.3. The algorithms are compared by the difference between the inverse of the average of their depth output for each trial and the inverse of the distance of the disks in that trial. Then a paired t-test is used to compare the errors, while outliers above 5 are ignored. In SQL the improved algorithm depth is calculated as shown in listing 6.1.

Listing 6.1: Calculation of improved algorithm depth

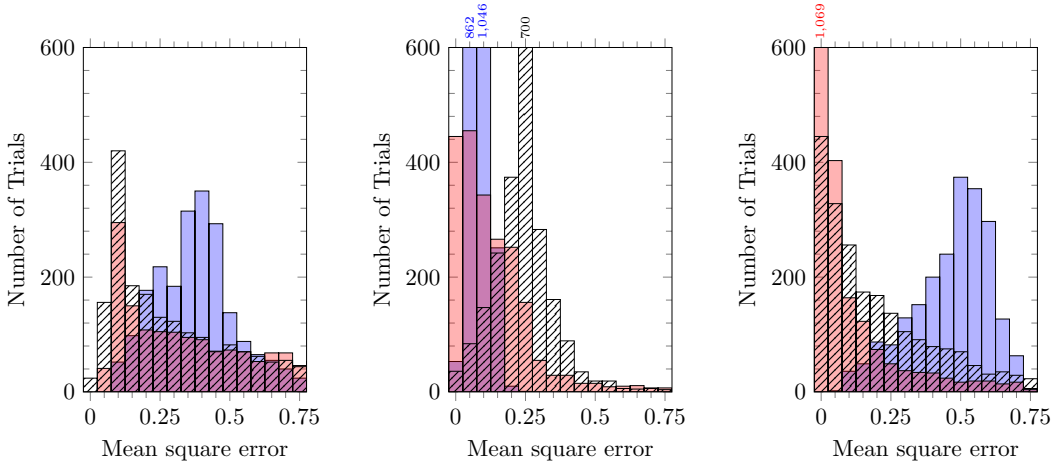
```
1 CASE WHEN r.vision_depth = -1
2      THEN 0.0
3      ELSE 1/SQRT(POW(r.vision_x,2)
4                  +POW(r.vision_y,2)
5                  +POW(r.vision_z,2))
6 END AS improved_vision_depth
```

The ray intersection point is calculated directly in SQL from the ray origins and directions saved in the database as the center point between the two closest points on the rays. The complete SQL code can be found on the accompanying DVD (see appendix B for details). Afterwards, the inverse distance is determined analogous to listing 6.1.

As the raycast can only be done with knowledge of the geometry, the raycast algorithm output is determined in Unity by reading each data set from the `recording` table, performing the raycasts for each eye and saving the resulting positions back into the table. The final inverse depth calculation is again analogous to the two values above.

6.1.1. Results

Figure 6.3 shows a histogram of the square errors for every trial and for each depth, comparing the three methods. For the close targets, the proposed method performs better than both reference methods (compared to ray intersection: $t = -11.8, p \ll 0.01$;



(a) close distance (b) middle distance (c) far distance

Figure 6.1.: Histogram of error distribution of estimate fixation targets using the proposed probability-based method, the ray intersection method, and raycast method for the (a) close, (b) medium, and (c) far targets

compared to raycast: $t = -21.9, p \ll 0.01$). For the middle targets, the proposed method performs worse than both baseline algorithms (compared to ray intersection: $t = 50.1, p \ll 0.01$; compared to raycast: $t = 5.67, p \ll 0.01$). For far targets, the proposed algorithm is better than the ray intersection method ($t = -40.5, p \ll 0.01$) but worse than the raycast method: ($t = 9.30, p \ll 0.01$). When trials from all depths are considered together, an overall improvement of the depth estimation compared to both comparisons is noticed (compared to ray intersection: $t = -1.63, p = 0.10$; compared to raycast: $t = -5.92, p \ll 0.01$).

6.2. Objective Evaluation of Fatigue

For the objective evaluation, it is important that the gaze estimation works. Figure 6.2 shows the mean square error using the probability method for each participant. Four participants upper quantile extends above 0.5, i.e. more than 25% of the trials had an average square error above 0.5. Therefore, these four participants were excluded from the analysis. No correlation between the results of the stereo acuity test and gaze estimation errors are noticed.

The participants completed 98.5% of all trials correctly (mismatching symbols were selected in only 96 out of 6681 trials). The average reaction time of one trial (i.e. the trial completion time) when dynamic adjustment was enabled (mean duration = $3.33s$, SD = $2.09s$) does not differ significantly from trials with fixed camera parameters (mean duration = $3.37s$, SD = $2.21s$). However, the average reaction time and most other measurements varied significantly between participants (see fig. 6.2). Therefore,

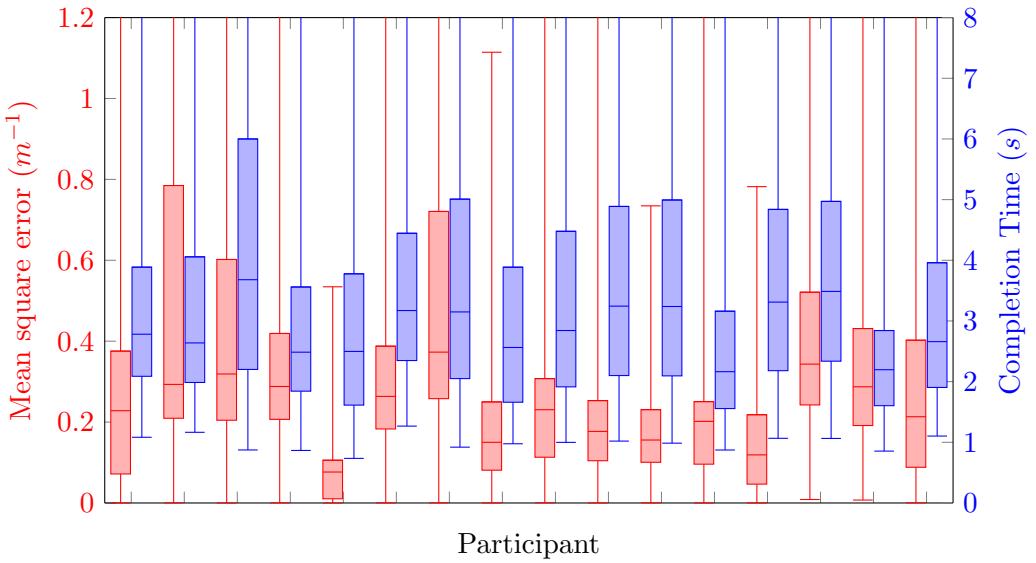


Figure 6.2.: Box plot of the distribution of mean square errors using the probability method and completion time for each participant.

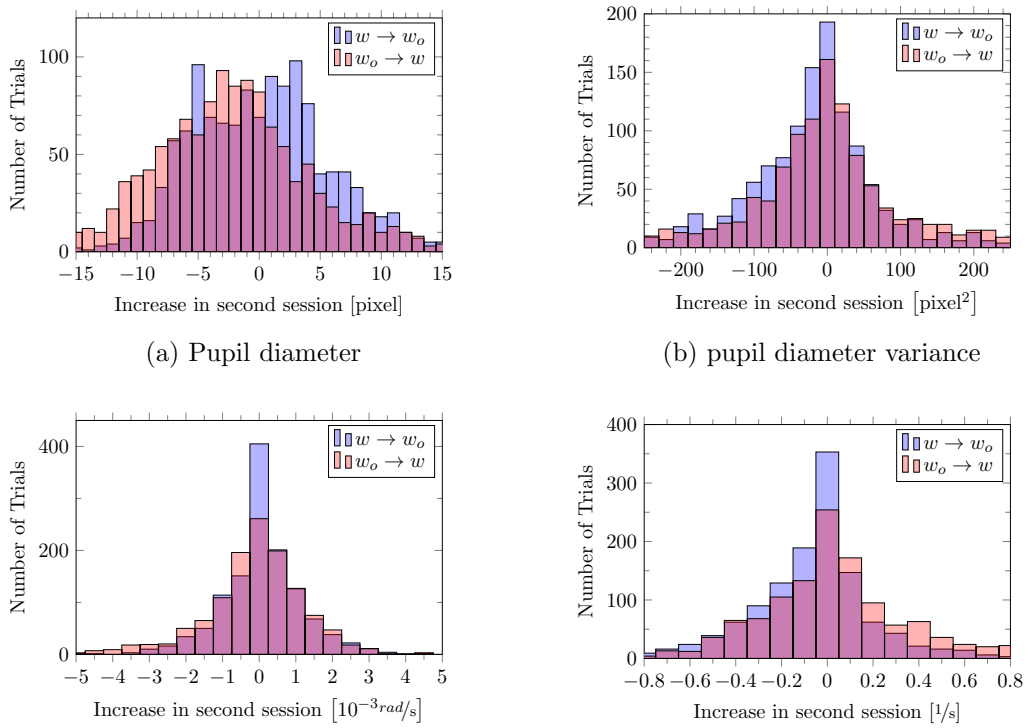
I have limited the analysis to inter-participant analysis. Most measurements also vary significantly between different trials. Therefore, the analysis is done by comparing the measurements between the two times each participant completed the same trial (once with and once without the adjustment method).

First, I used the periodically recorded measurements (see section 5.5.2) to calculate comparable aggregate measurements per trial and session:

- pupil diameter are averaged
- pupil diameter variance is calculated
- distances between eye-positions are added up and divided by the trial duration to calculate the average eye movement speed during the trial.

Furthermore, the time difference between start and end time was calculated to determine the total reaction time. As the reaction time is log-normal distributed, the logarithm of the reaction time was calculated. Finally, the blink rate was calculated as amount of blinks in a time window of 10 seconds centered in the middle of the trial divided by 10. All of these parameters can be related to visual fatigue (see results). The calculation was done using the SQL View `experiment_parameters` that can be found on the accompanying DVD (see appendix B for details).

Then I computed the trial-wise differences of each measurement and participant yielding a positive value when the measurement is larger in the second session and a negative value when it is smaller. Finally, the distributions of the differences were compared between the two groups of participants (based on the order of sessions) using a paired



(a) Pupil diameter

(b) pupil diameter variance

(c) eye movement speed

(d) blink rate

Figure 6.3.: Histograms of the amount of participant's trial execution where each specific difference of measurements is observed, differentiated by group of the participant (see the legends). w corresponds to the sessions when dynamical adjustment was enabled and w_o corresponds to the sessions of no adjustment.

t-test. Some change between the sessions is expected as fatigue might always be more prominent in the second session and participants got more used to the game. However, differences between the groups would indicate that the adjustment has an influence on the measurements.

6.2.1. Results

Significant changes can be found for the pupil diameter differences ($t = 10.15; p \ll 0.01$), eye movement speed differences ($t = 3.86; p \ll 0.01$), and blink rate differences ($t = -15.01; p \ll 0.01$). Pupil diameter variance also shows some difference, but not significantly ($t = -0.86; p = 0.39$). Figure 6.3 shows the histograms of changes for these four measurements. The differences of the logarithm of the reaction time ($t = 0.13; p = 0.89$) do not show a difference between the groups.

The pupil diameter decreased on average in the second session in both groups. However, it decreased by only 0.0099 pixel for the group that had the adjustment method first (group 1) and by 1.29 pixel for the group that had the adjustment method second (group 2). This indicates that the adjustment method has a negative impact on pupil diameter. A smaller pupil diameter is believed to indicate increased visual fatigue [29].

The pupil diameter variance is smaller on average in the sessions done without the adjustment for both groups. It is smaller by 0.30 pixel^2 in group 1 and by 1.74 pixel^2 in group 2. A larger pupil diameter variance indicates more fatigue [29].

The eye movement speed is larger on average in the sessions done without the adjustment for both groups. It is increased by $0.97 \cdot 10^{-4} rad/s$ in group 1 and by $1.22 \cdot 10^{-4} rad/s$ in group 2. While eye movement speed is not a common measurement for fatigue, a slower eye movement speed is likely to be an indicator for fatigue.

The blink rate is smaller on average in the sessions done without the adjustment for both groups. It is smaller by $1.49 \cdot 10^{-2} 1/s$ in group 1 and by $1.64 \cdot 10^{-2} 1/s$ in group 2. A higher blink rates indicates higher fatigue [30].

The log reaction time decreases for group 1 and group 2 in the second session by 0.08 and 0.10 respectively. While no significant difference between the groups is noticed, this indicates that participants of both got more used to the game and thus were better in the second session.

As detailed, all measured parameters, where significant differences between the groups were detected, indicate that the adjustment method increases discomfort.

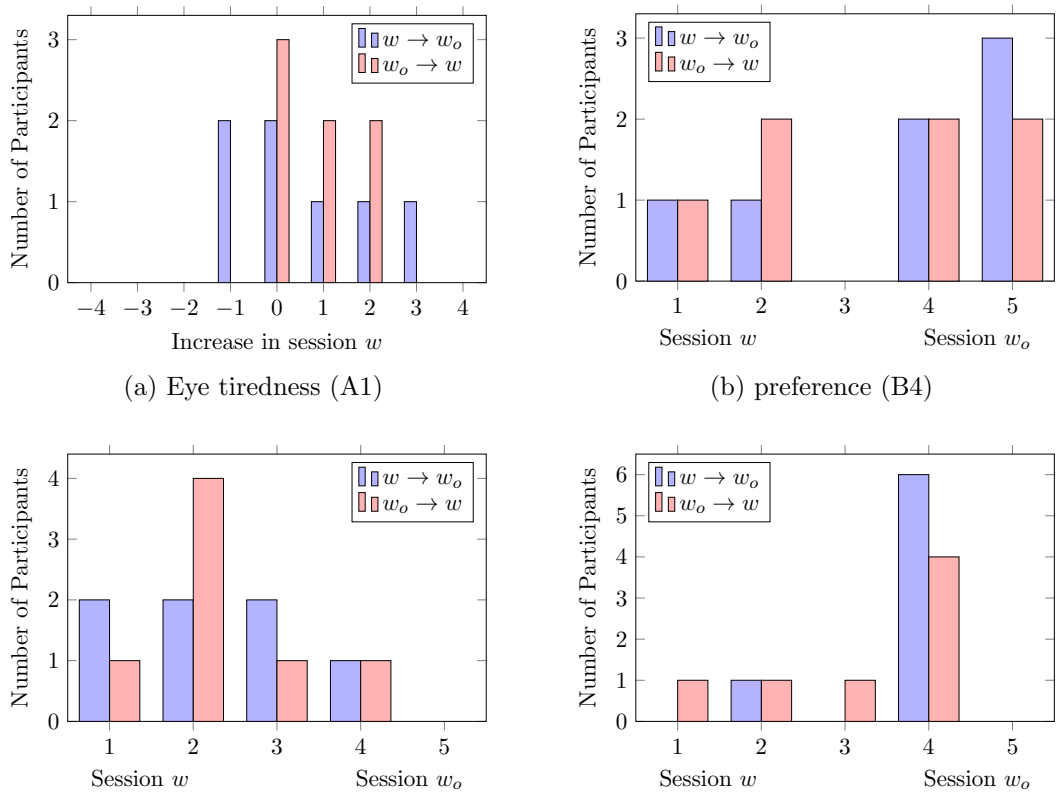
6.3. Subjective Evaluation of Fatigue

The questionnaires are evaluated similarly to the objective measurements. For the questions in block A the differences of the answers for each session of each participant were calculated. These differences are then compared between the two groups of participants. The answers for the questions in block B were directly compared between the groups. The participants excluded from the objective evaluation were also excluded from the subjective evaluation.

6.3.1. Results

No participant correctly identified the difference between the sessions. Furthermore, no participant noticed any camera movement during either session.

The answers to most of the other questions asked do not show a clear trend. Figure 6.4 shows histograms for four of the questions asked that do show a trend. Here, the answers of group 2 were flipped to simplify their reading. Thus, answers to the right represent



(a) Eye tiredness (A1)

(b) preference (B4)

(c) eye irritation (B2)

(d) easier session (B5)

Figure 6.4.: Subjective responses to the questions indicated in brackets, differentiated by group of the participant. w: sessions with dynamic adjustment; w_o: the sessions without adjustment. (a) Difference in answer in the session with adjustment to the session without adjustment; (b)-(d) answers to questions, flipped for participants $w_o \to w$. (A1): question asked (see section 5.6)

the session with the adjustment and not the second session. The histograms indicate, that eye tiredness and eye irritation might be increased in the session with adjustment (see fig. 6.4a and 6.4c). Also, most participants found the game easier to play in the session without adjustment (see fig. 6.4d) and a slight preference for that session can be observed (see fig. 6.4b).

While the amount of participants is not sufficient to determine any statistically significant results, these trends go into the same direction observed in the objective evaluation.

6.4. Discussion

The results show that an adjustment of the camera separation and vergence does not yield reduced fatigue. Instead, such an adjustment increases the fatigue. This result goes in line with previous findings [10] that improvements in visual comfort achieved by algorithm solutions are limited and it is difficult to effectively reduce the vergence-accommodation conflict unless physical changes of accommodation is involved. Some related work [11], [12] has reported improvement using a dynamic camera adjustment. However, they do not use an eye tracker for the depth information but instead tell the participants where to look or use a very simple prediction. This does cause the depth information to almost always be correct but their real world applications are limited. This shows that a significantly improved eye tracking depth estimation might make an improvement using the adjustment method possible.

Of the two dimensional space of variable adjustments in the algorithm, this thesis only considered a subspace by following a simple protocol as described in section 4.3. It remains unclear how other combinations of adjustments, for example by allowing camera translation and rotation at the same time, would affect visual comfort. An analysis of different protocols might be interesting.

Furthermore, the experiment consisted of a static scene. While the focus depth was changed during the experiment, the individual targets remained stationary, such that no pursuit eye-movement was necessary. It is possible, pursuit movements with changing depth cause especially strong discomfort when the eye accommodation remains constant. Therefore, the adjustment method might be beneficial in those cases.

An adjustment similar to the one proposed in this thesis can also be used with different motivation. For example, camera adjustments can yield better depth perception and user preference [12], [17], [18]. This thesis did not evaluate depth performance objectively. While subjective depth perception was asked in the questionnaire (question B8), participants were not told to pay attention to this. Therefore, the answers to this question were inconclusive. The reduction in visual fatigue reported is only small, such that it should not pose a serious issue, if the adjustment is used for different reasons.

In contrast to the adjustment method, the eye tracking depth estimation algorithm does yield an improvement in accuracy compared to simple ray intersection and raycasting algorithms. The optimization of the algorithm is only done based on few data. A personal optimization for each user might improve its performance. Also, this system requires an accurate depth map of the entire field-of-view from both eye perspectives. Therefore, its application outside HMDs is limited. Also, there are only a small amount of use-cases that require accurate depth estimation in addition to positional information. Furthermore, the depth estimation algorithm is computationally relatively expensive. Moving the calculation to the GPU could significantly improve its performance. Lastly, if the object focused at is only visible to one eye, all of the depth estimation algorithms fail. Some data from the optimization in section 3.4.3 shows that the eyes might converge to the distance of the obstructing object instead of the focused object distance. Further research is needed to detect this condition and properly react to it.

7. Conclusion

The vergence-accommodation-conflict poses a significant problem when designing stereoscopic 3D media. In traditional applications it is usually solved by limiting the depth to an acceptable range. While visual discomfort is mostly resolved with this solution, the limited depth range reduces the possible 3D effect. The increasingly popular HMDs suffer from the same problem.

Much research has been done trying to solve the vergence-accommodation-conflict. Some propose to use artificial blurring to fake the eyes depth of field and some try to use monovision where the two eyes focus at different depth. However, most software approaches have been shown to be ineffective. Physical dynamic adaptation of the HMD optics to change the eyes focus distance do work, but are inconvenient for consumer HMDs. In this thesis, a system to change the vergence to match the accommodation has been tested in combination with depth estimation based on an eye-tracker.

First a probability based algorithm to determine the gaze-depth has been proposed. This algorithm accounts for the geometry of the scene and the measured eye convergence. The algorithm models each eyes view direction as a normal distribution around the measured direction and determines the point on the scene geometry that has the highest probability. The algorithm also takes into account the missing independence between the two normal distributions by using an additional normal distribution based on the difference between the measurements of each eye. The ratio between the two standard deviations is calibrated using a simple experiment.

The dynamic adjustment of the vergence is done using the two camera movements divergence and separation. Camera distance has been shown to be harmfull when adjusting for vergence too far and camera convergence has been shown to cause confusion when adjusting for vergence too close. Therefore, a simple protocol is used where separation is used for close objects while divergence is used for far objects.

An experiment was conducted to determine the effectiveness of the adjustment and the quality of the eye tracking depth estimation. The experiment was based on the *Dobble* game. Participants completed two sessions while objective measurements were done. After each sessions they were asked to evaluate their fatigue in a questionnaire.

Results show that the probability based algorithm to determine the gaze depth is better than the baseline algorithms in general, but that it is worse for a medium distance. Results also show that the adjustment method increases discomfort and fatigue. This underlines that physical adaptation of HMD optics is the only feasible method to solve the vergence accommodation conflict. However, it is possible that a further improvement of the estimation of the gaze depth or a different adjustment protocol produces improvements. For now, it is advisable to refrain from using such an adjustment system as it do not produce the desired result.

Bibliography

[1] W. J. Tam, F. Speranza, S. Yano, K. Shimono, and H. Ono, "Stereoscopic 3D-TV: Visual Comfort", *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 335–346, Jun. 2011, ISSN: 0018-9316. DOI: 10.1109/TBC.2011.2125070.

[2] T. E. Lockhart and W. Shi, "Effects of Age on Dynamic Accommodation", *Ergonomics*, vol. 53, no. 7, pp. 892–903, Oct. 2010, ISSN: 0014-0139. DOI: 10.1080/00140139.2010.489968.

[3] C. M. Schor, L. A. Lott, D. Pope, and A. D. Graham, "Saccades reduce latency and increase velocity of ocular accommodation", *Vision Research*, vol. 39, no. 22, pp. 3769–3795, Oct. 1999, ISSN: 0042-6989. DOI: 10.1016/S0042-6989(99)00094-2.

[4] S. Shiwa, K. Omura, and F. Kishino, "Proposal for a 3-D display with accommodative compensation: 3DDAC", en, *Journal of the Society for Information Display*, vol. 4, no. 4, pp. 255–261, Oct. 1996, ISSN: 1938-3657. DOI: 10.1889/1.1987395.

[5] W. Chen, J. Fournier, M. Barkowsky, and P. Le Callet, "New stereoscopic video shooting rule based on stereoscopic distortion parameters and comfortable viewing zone", en, in *IS&T/SPIE Electronic Imaging*, A. J. Woods, N. S. Holliman, and N. A. Dodgson, Eds., vol. 7863, Feb. 2011, 78631O. DOI: 10.1117/12.872332.

[6] N. Padmanaban, R. Konrad, T. Stramer, E. A. Cooper, and G. Wetzstein, "Optimizing virtual reality for all users through gaze-contingent and adaptive focus displays", en, *Proceedings of the National Academy of Sciences*, p. 201 617 251, Oct. 2017, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1617251114.

[7] P. V. Johnson, J. A. Parnell, J. Kim, C. D. Saunter, G. D. Love, and M. S. Banks, "Dynamic lens and monovision 3D displays to improve viewer comfort", *Optics Express*, vol. 24, no. 11, p. 11 808, May 2016, ISSN: 1094-4087. DOI: 10.1364/OE.24.011808.

[8] R. Konrad, E. A. Cooper, and G. Wetzstein, "Novel Optical Configurations for Virtual Reality: Evaluating User Preference and Performance with Focus-tunable and Monovision Near-eye Displays", ser. CHI '16, ACM, Oct. 2016, pp. 1211–1220, ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858140.

[9] Y. Yadin, S. Rosen, Y. Haddad, Y. Vardi, and I. Grutman, "Invited Paper: Resolving the Vergence Accommodation Conflict in VR and AR via Tunable Liquid Crystal Lenses", en, *SID Symposium Digest of Technical Papers*, vol. 49, no. 1, pp. 870–873, Oct. 2018, ISSN: 2168-0159. DOI: 10.1002/sdtp.12272.

[10] G.-A. Koulieris, B. Bui, M. S. Banks, and G. Drettakis, "Accommodation and Comfort in Head-mounted Displays", *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–87, Oct. 2017, ISSN: 0730-0301. DOI: 10.1145/3072959.3073622.

[11] E. Peli, T. R. Hedges, J. Tang, and D. Landmann, "A Binocular Stereoscopic Display System with Coupled Convergence and Accommodation Demands", en, *SID Symposium Digest of Technical Papers*, vol. 32, no. 1, p. 1296, Oct. 2001, ISSN: 0097966X. DOI: 10.1889/1.1831799.

[12] A. Sherstyuk, A. Dey, C. Sandor, and A. State, "Dynamic Eye Convergence for Head-mounted Displays Improves User Performance in Virtual Environments", ser. I3D '12, ACM, Oct. 2012, pp. 23–30, ISBN: 978-1-4503-1194-6. DOI: 10.1145/2159616.2159620.

[13] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, M. Gross, M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross, "Nonlinear disparity mapping for stereoscopic 3D", in *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10*, vol. 29, New York, New York, USA: ACM Press, 2010, p. 1, ISBN: 9781450302104. DOI: 10.1145/1833349.1778812.

[14] A. T. Duchowski, D. H. House, J. Gestring, R. I. Wang, K. Krejtz, I. Krejtz, R. Mantiuk, and B. Bazyluk, "Reducing Visual Discomfort of 3D Stereoscopic Displays with Gaze-contingent Depth-of-field", ser. SAP '14, New York, NY, USA: ACM, Oct. 2014, pp. 39–46, ISBN: 978-1-4503-3009-1. DOI: 10.1145/2628257.2628259.

[15] S. A. Cholewiak, G. D. Love, P. P. Srinivasan, R. Ng, and M. S. Banks, "Chromablur: rendering chromatic eye aberration improves accommodation and realism", en, *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–12, Oct. 2017, ISSN: 07300301. DOI: 10.1145/3130800.3130815.

[16] M. Mauderer, S. Conte, M. A. Nacenta, and D. Vishwanath, "Depth Perception with Gaze-contingent Depth of Field", ser. CHI '14, ACM, Oct. 2014, pp. 217–226, ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557089.

[17] M. Fisker, K. Gram, K. K. Thomsen, D. Vasilarou, and M. Kraus, "Automatic convergence adjustment for stereoscopy using eye tracking", The Eurographics Association, 2013. DOI: 10.2312/conf/EG2013/posters/023-024.

[18] P. Kellnhofer, P. Didyk, K. Myszkowski, M. M. Hefeeda, H.-P. Seidel, and W. Matusik, "GazeStereo3D: Seamless Disparity Manipulations", *ACM Trans. Graph.*, vol. 35, no. 4, 68:1–68:13, Oct. 2016, ISSN: 0730-0301. DOI: 10.1145/2897824.2925866.

[19] A. T. Duchowski, *Eye Tracking Methodology*, Second Edi. Clemson: Springer International Publishing, 2017, ISBN: 978-3-319-57881-1. DOI: 10.1007/978-3-319-57883-5.

[20] M. Kassner, W. Patera, and A. Bulling, "Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-based Interaction", Apr. 2014.

[21] F. Mulvey, "Woking copy of definitions and terminology for eye tracker accuracy and precision", 2010.

[22] E. Gutierrez Mlot, H. Bahmani, S. Wahl, and E. Kasneci, "3D Gaze Estimation Using Eye Vergence", 2016, pp. 125–131.

[23] J. Geng, *Three-dimensional display technologies*, 4. NIH Public Access, 2013, vol. 5, pp. 456–535. DOI: 10.1364/AOP.5.000456.

[24] A. Woods, T. Docherty, and R. Koch, "Image Distortions in Stereoscopic Video Systems", J. O. Merritt and S. S. Fisher, Eds., vol. 1915, International Society for Optics and Photonics, Sep. 2002, pp. 36–48. DOI: 10.1117/12.157041.

[25] W. R. Watkins, "Enhanced depth perception using hyperstereo vision", W. R. Watkins and D. Clement, Eds., vol. 3062, International Society for Optics and Photonics, Jun. 1997, pp. 117–125. DOI: 10.1117/12.276688.

[26] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder, "Foveated 3D Graphics", *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–164, Oct. 2012, ISSN: 0730-0301. DOI: 10.1145/2366145.2366183.

[27] Q. Sun, A. Patney, L.-Y. Wei, O. Shapira, J. Lu, P. Asente, S. Zhu, M. Mcguire, D. Luebke, and A. Kaufman, "Towards Virtual Reality Infinite Walking: Dynamic Saccadic Redirection", *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–67, Oct. 2018, ISSN: 0730-0301. DOI: 10.1145/3197517.3201294.

[28] O. V. Komogortsev and J. I. Khan, "Eye movement prediction by Kalman filter with integrated linear horizontal oculomotor plant mechanical model", in *Proceedings of the 2008 symposium on Eye tracking research & applications - ETRA '08*, New York, New York, USA: ACM Press, 2008, p. 229, ISBN: 9781595939821. DOI: 10.1145/1344471.1344525.

[29] Y. Morad, H. Lemberg, N. Yofe, and Y. Dagan, "Pupillography as an objective indicator of fatigue", *Current Eye Research*, vol. 21, no. 1, pp. 535–542, Jan. 2000, ISSN: 0271-3683. DOI: 10.1076/0271-3683(200007)2111-ZFT535.

[30] J. A. Stern, D. Boyer, and D. Schroeder, "Blink Rate: A Possible Measure of Fatigue", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 36, no. 2, pp. 285–297, Jun. 1994, ISSN: 0018-7208. DOI: 10.1177/001872089403600209.

A. Custom Stereo Camera Rig in Unity

When using Unity3d with an HTC Vice, all cameras that are set to render to the HMD will automatically be tracked such that they correspond to the HMD location. The camera field of view, camera separation and camera convergence are automatically set to correspond to the HMD. For usual use cases this simplifies the development for HMDs and generalizes all HMDs such that the developer does not need to consider every HMD model individually. Unfortunately, this feature can not be disabled. Therefore, I needed to use a relatively complicated custom camera rig as shown in fig. A.1. I use one camera for each eye, set such that they only render to their corresponding side in the HMD. To be able to control their separation I first negate their automatically tracked position by nesting them in objects whose local position is the negative of the HMD position (`NegateTracking`), so that the global camera positions do not move anymore. This is done using a simple script with the update function shown in listing A.1. In the script `negateNode` is a `XRNode` corresponding to the eye they should negate.

Listing A.1: Update function of the `NegateTracking` script

```
1 void Update(){
2     transform.localPosition =
3         -InputTracking.GetLocalPosition(negateNode);
4     transform.localRotation = Quaternion.Inverse(
5         InputTracking.GetLocalRotation(negateNode));
6 }
```

In order to move these negation nodes, they themselves are nested in empty objects (`LeftEye` and `RightEye`). Finally the empty object for each eye are nested in a common parent object (`StereoCameraRig`) that handles their local positioning and is tracking the hmd position.

Figure A.1.: Tree-view of stereo camera rig

B. Index of Accompanying DVD

This appendix gives an overview over the file structure of the accompanying DVD.

```
DVD
├── README.txt    Readme file containing this index
├── database.sqlite    Database as described in section 5.4
├── referenced
│   ├── optimization_recordings    see section 3.4.3
│   │   └── *.csv
│   ├── pupil_plugin_accuracy.py    see section 3.2
│   ├── ConvergenceDepth.cs
│   ├── ConvergenceDepthMP.cs    see section 3.4.1
│   ├── RayIntersectionView.sql    see section 6.1
│   └── ExperimentParametersView.sql    see section 6.2
├── unity_project    Unity project used
│   └── *
└── thesis.pdf    PDF version of this thesis
```