# Searching for the correlation between the pupil size, the accommodation and the vergence of the eyes using mental tasks and 2D stimuli

Stefan Jansen 4293290
Julia Russell 4451538
Tamir Themans 4471687

June 2019

## 1 Abstract

**Background.** The pupil size is an important index of the mental workload or excitement. In pupillometry, confounding factors, such as accommodation or vergence, are usually not taken into account. This study focuses on whether there is a correlation between pupil diameter, accommodation and vergence when performing mental effort or viewing 2D stimuli.
**Methods.** The pupil diameter and refraction were measured simultaneously while participants had to solve multiplications. The multiplications were divided into three levels of difficulty. In addition to solving multiplications, the participants looked at static and dynamic stimuli. Here, the gaze shift was measured as well. The static and dynamic stimuli were shown in four different images, on a scale from abstract to realistic. The more realistic an image was, the more depth cues were available in that image.
**Results.** The pupil dilated when performing cognitive tasks. However, the magnitude of accommodation was unaffected by the cognitive tasks. Furthermore, results showed that it does not make a difference for the amount of change in pupil diameter, accommodation or gaze(Y) if a stimulus is static or dynamic. Also, the level of realism of a stimulus did not change the amount of change in pupil size, accommodation or gaze.
**Conclusion.** Non of the findings indicated a correlation between the pupil size, the accommodation and the gaze shift when viewing 2D stimuli, or without gaze shift when performing mental effort. Prior studies that made use of pupillometry on a 2D screen will not be judged on the fact that they did not take accommodation and vergence into account while measuring the pupil size.

# 2 Introduction

Pupillometry, the measurement of the eye pupil response to stimuli, is an important research tool in psychological studies. For example, early studies have indicated that the pupil size provides information about mental workload [14], [15] and interest [8], [7], [14]. This given, researchers use pupil size as an indication for the amount of work load or excitement participants in their studies encounter.
Researchers focus on measuring pupil size, but do not take confounding physiological factors into account. From just determining the pupil size, without taking other variables of the eye in consideration, wrong conclusions could be drawn. The pupil regulates the amount of light that reaches the retina. The pupil size is measured in millimeters (mm) [3] [17].

A possible confounding physiological factor that is not often considered in pupillometrics research is accommodation, that is, the changing of the thickness of the eye lens. Its function is to create a sharp picture on the retina [10], [17]. It is measured in Dioptres (1/m).

The pupil size could be correlated to the accommodation of the lens, but it could also be correlated to the vergence of the eye. Vergence occurs when eyes make rotational movements so that an object is projected in the centre of the retina in both eyes. When looking at an object close by, the eyes rotate towards each other (convergence). When looking at an object further away, the eyes rotate away from each other (divergence) [16]. Vergence occurs when there is shift gaze from targets at a different distance. The vergence can be measured as a combination of the gaze(X) from both eyes. Gaze(X) is the angle of vision over the x-axis (looking left and right) and gaze(Y) is the angle of vision over the y-axis (looking up and down) [10], [3].

Together, the pupil size, the vergence and the accommodation form a triad. To learn more about this triad and the interaction between these three, research needs to be done.

## 2.1 Prior research

The main purpose of the pupil is to regulate the light that enters the eye: the pupil constricts as a result of brightness and dilates in darkness, what is called the pupillary light response. The pupil also constricts when the eye is fixated on stimuli nearby. This is called the near response or accommodation reflex. Another well-established trigger of pupil response is its dilation during mentally demanding tasks [15]. One of the first studies that showed that the pupil dilates during mental activity was conducted in 1964 by Hess and Polt [9], which was confirmed by several other studies [14], [11], [13].

Enright (1987) investigated the interaction between pupil size and vergence. He used different stimuli at a fixed distance. Perspective drawings containing depth cues were looked at with monocular vision and real objects were looked at with monocular and binocular vision. Changing the monocular focus of the eye from the 'front' to the 'back' of perspective drawings several times resulted in convergence and divergence of the eye, even though the object was in a single plane. The magnitude of the reaction depended on the object shown. The pupil did not act as Enright expected. He expected constriction of the pupil when focusing on the the 'front' of the perspective drawing a result of the pupillary near response. However, the pupil dilated when the eyes converged [3].

Other studies conducted research on the pupil size and accommodation of the eye. Observed was that by doing simple mathematical tasks, which involves adding, multiplication or reading, a correlation between pupil size and mental

workload was found. Cognitive demand and Dioptres of the lens did not correlate [14], [11]. Jainta (2008) researched this as well. The aim of the study was to ensure if closed-loop accommodation indicates cognitive induced changes in autonomic balance. Next to this, gaze shift and performance measures were taken into account. Just like the other studies, a correlation between cognitive demand and accommodation could not be measured. Recommendations were done to work more with closed-loop accommodation in further studies [12].
What is meant by open-loop in this paper, is that the test subject does not have to focus at an object. By doing so, the test subject will go into a 'stand-by modus'. The focus position will not become zero dioptric or infinity, but the eyes will converge to a specific point in the distance [10] [18]. This means that with a closed-loop experiment, the movement of the eye can be measured best.

Feil, Moser and Abegg (2017) did not focus on cognitive demand but on gaze shifts from a far to a near object. They stated that if eyes need to focus from an object in the distance to something nearby, it leads to retinal image disparity and blur. A vergence response is triggered by this disparity and accommodation is triggered by the blur. The accommodation and vergence reaction are dependent. They suggested that the reaction of the pupil depended mostly on convergence and not on accommodation. The near triad depends on the vergence system [4].
Still, there is much unknown about how the the pupil size, the accommodation of the lens and the vergence influence each other.

## 2.2 Goal of current research

The purpose of this study is to learn how the accommodation, the vergence and the pupil diameter are connected to one another.
The following research question has been formulated:
Is there any direct correlation between the pupil size, the accommodation and the vergence of the eyes when looking at 2D stimuli and performing mental effort?

In order to examine this question, three subquestions are composed:
1. Does mental effort (performing multiplications) result in a change in accommodation or an increase in pupil diameter?
2. Does it make a significant difference to the amount of accommodation, gaze shift or change in pupil size if the movement on a 2D screen is static instead of dynamic?
3. Does the number of depth cues that a static or dynamic 2D-stimuli contains, makes a difference in the way accommodation, the gaze shift and pupil size change?

Depth cues are visual means to perceive an image in 3D when it is actually in 2D, so they create perception of depth. The more depth cues are added to a stimuli, the more depth the mind perceives. One example of a depth perception is occlusion: When an object in the front overlaps an object that is more in the distance, this creates depth [6], [5]. A complete overview of all depth perceptions is shown in Appendix A.

## 2.3 Hypotheses

Based on the prior research, hypotheses could be formed for the three sub-questions.
1. There will not be a change in accommodation caused by performing cognitive tasks. The pupil will dilate during mental effort and constrict again after the effort is done.
2. No prior research conducted the difference between static and dynamic stimuli, so no hypotheses can be formed for this question.
3. The more depth cues present, the more depth is perceived, so the stimuli seems more realistic to the mind. Therefore, more accommodation will occur, when more depth cues are present.

The magnitude of the gaze shift does not depend on the amount of depth cues in a stimuli. The pupil diameter does not depend on the amount of depth cues in a stimuli.

From prior research it is hard to form a hypothesis for the research question. It is known that in the pupillary near response there is a connection between the pupil size, the accommodation and the vergence. When a object moves towards the eyes, the vergence increases, the refraction of the lens increases and the pupil constricts [15], [17].

When performing cognitive tasks, the expectation is that the pupil dilates but the accommodation will not change. So there is no correlation between the accommodation and pupil size when doing mental effort.

Expected is that there is no correlation between the pupil size, the accommodation and the vergence when looking at 2D stimuli.

# 3 Methods

## 3.1 Participants

In total, 35 participants took part in this research (20 males, mean age = 22.0 years; standard deviation (SD) = 1.59). Every participant signed a written consent form [Appendix B.1].

## 3.2 Measuring device

The tests were done with the PowerRef III (Plusoptix) [figure 1]. This device made it possible to measure the accommodation, the pupil size and the Gaze(X and Y) at the same time with binocular vision. As explained earlier the most studies in the past used monocular vision [3], [12]. Measuring both eyes at the same time made it interesting to take closed-loop into account, as opposed to open-loop.
The experimental set-up was used in the following manner: The participants rested with their chin on a mount, so that their eyes were stable and kept at the same place. The participants could see the computer screen (BenQ, XL2420Z) through a hot mirror (transparent) [2]. The camera from the PowerRef III measured the eyes of the participants through the reflected light from the hot mirror and the tiled mirror. The participants could see the whole screen without the limitations of having the PowerRef III between the eyes and the screen. The PowerRef III was running independent of the presentation computer (see table 1 for specifications) running the MATLAB code [Appendix C.1 and Appendix C.2] that displayed the stimuli on the BenQ computer screen. To synchronize the measurements of the PowerRef III to the visuals send from the computer with the MATLAB-code, pulses were sent from the presentation computer via an NMC cable to the PowerRef III. These pulses were encoded in the data of the measurements. On top of that, multiple clock times were saved during the execution of the Matlab code. This was done to check timing of the pulses and to determine how long the computer needed to execute the whole code. The PowerRef III measures with 50 Hz and therefore it is essential to make sure the pulses sent to the PowerRef III are not a few samples off. The data collected by the PowerRef III was put in a CSV-file automatically, which was used for off-line data analysis.

*Table 1: Specifications devices*

| Device | Specification |
|---|---|
| Presentation computer | CPU: Intel(R) Core$^{TM}$ i7-6700 CPU @3.4 GHz<br>RAM: 16 GB Single Channel @ 1064 MHz<br>MOBO: MSI H110M Pro-D (MS-7996)<br>GPU: NVIDIA GeForce GTX 1070 4GB<br>Storage: 500 GB Samsung SSD 850 EVO (SATA SSD)<br>1 TB Toshiba DT01ACA100 (SATA) |
| Computer screen | BenQ XL2420Z |


*Figure 1: Set-up of the experiment with the PowerRef III*

## 3.3 Stimuli

The experiment consisted of three different tasks. Throughout the whole experiment the background was grey. This grey contained the

RGB values [127 127 127]. During all experiments, colours were eliminated because their presence will influence the three parameters in the triad.

**Task 1:** During the first task, the participants listened to two different tones while performing an eye movement task. The high pitched tone was 1000 Hz. The low pitched tone was 100 Hz. A wooden object -a plate with two sticks attached on top of it- was placed between the PowerRef III and the screen. The screen showed the same stimuli through task 1 [Appendix E.1]. The eyes had to change their focus to the right stick, which was 40.608 centimetres away from the eye when hearing a high tone and to the left stick, which was 80.306 centimetres away from the eye, when hearing a low tone [see Appendix D.1 for the setup]. These dimensions are not completely rounded up. This is due to the fact that the sticks are not placed in line with the bridge of the eyes, but a little bit to the right and to the left of the centre line. The reason for this placement is to make it absolutely clear to the participants which stick to look at.

**Task 2:** During the second task the participants had to solve nine multiplications [Appendix F]. The nine multiplications were divided into three parts: easy, average and difficult multiplications (table 2). The mean of three multiplications in one part were considered as a result. Before every multiplication a control slide was shown [Appendix E.2].

*Table 2: Nine multiplications of task 2. Divided in three subgroups, making a distinction between the difficulties; easy, average and difficult.*

| Easy | Average | Difficult |
|---|---|---|
| 6x12 | 8x16 | 16x18 |
| 7x14 | 9x14 | 15x16 |
| 8x13 | 11x13 | 14x17 |

**Task 3:** During the third part, four tests were done, all four containing a static and a dynamic part. So the participants got to see eight different kind of videos. The four test were on a scale from very abstract to more realistic. The more realistic the stimuli was supposed to be, the more depth cues were added. This was done to make the illusion of depth more realistic. During all four tests a ball travelled over a road into the distance and back. To not be reliant on one version of this loop, this sequence was repeated three times. This results in a total amount of 24 videos.

- Level 1: no road
  This level showed a ball moving without any background [Appendix G.1]. One depth cue, depth from motion, created depth in this image.
- Level 2: simple road
  The simple road creates depth by making use of depth from motion and linear perspective. A simple road with no scenery (objects beside the road) was shown [Appendix G.2].
- Level 3: average road
  The average road has trees alongside the same road [Appendix G.3]. The trees become smaller and are placed higher as they are more in the distance. This added the the depth cues: relative size and relative height. Lines were added on the road, this added the depth cue texture gradient.
- Level 4: realistic road
  The most realistic road has trees and buildings placed next to the same road [Appendix G.4]. The buildings and trees are placed in front of each other and the more in the distance the tree or building is, the less details they have. This added two depth cues: occlusion and atmospheric perspective.

Before every one of the 24 videos, a control slide was shown [Appendix E.3]. This slide contained the start position of the ball on level 1. After 10 seconds, the videos started to play wit a duration of six seconds. After the 24 videos the whole experiment was finished.

A. Static stimuli
An image of the road with a ball in the beginning of the road was shown. After one second a new image showed up, now the ball is in the middle of the road. Again after one second the ball is shown in the back of the road. Two seconds later the ball moved back to the middle and then again after one second to the beginning of the road. This was repeated for all four levels of roads.

B. Dynamic stimuli
The ball moved smoothly from the beginning to the back and back to the beginning of the road again. This was repeated for all four levels of roads.

## 3.4 Experimental design

After an inform consent form [Appendix B.1] was signed and a questionnaire [Appendix B.2] was filled in, the participants seated themselves in front of the PowerRef III. The BenQ computer screen was placed at one meter distance from the eye, so the resting accommodation should be 1 D. Participants were given a set of noise cancelling headphones (DT 770 Pro, Beyerdynamic) to wear during the entire experiment.

### 3.4.1 Task 1

A wooden object is placed between the PowerRef III and the screen. Two sticks, marked with a black dot, are visible for the participant [Appendix D.1]. Prior to starting the task, instructions were shown on the upper half of the screen [Appendix H.1]. Participants were instructed to look at the left stick and keep their focus on it once a low tone was presented to them, and to look at the right stick and keep their focus on it once a high town was presented to them. Once the participants understood everything that was stated on the instruction slide, they pressed space bar on the keyboard that was located to the left of them. This automatically started the task. Between the presentation of each tone, there was a pause of ten seconds. In total three low and three high pitched tones were presented to the participants as can be seen in Figure 2. A more detailed overview of task 1 can be found in Appendix D.2. Overall task 1 lasted 60 seconds.

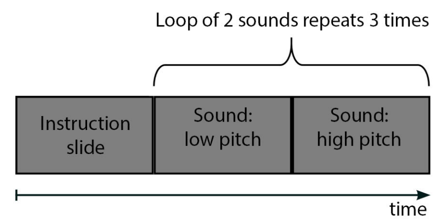


*Figure 2: Overview task 1*

### 3.4.2 Task 2

At the start of this task an instruction slide was shown to the participants [Appendix H.2]. Participants were instructed to mentally solve multiplications. Once the participant read and understood what was detailed on the instruction slide, they pressed the space bar and initiated the second task. Before every multiplication, a control slide was shown [Appendix E.2]. The control slide remained on the screen for ten seconds, after which a multiplication appeared. Participants had 15 seconds to solve the multiplications. Once they solved the multiplication, they had to press space bar and then audibly state the answer. The multiplication remained visible for the remaining time the participant had left of the 15 seconds before the control slide would appear again and the loop started over as can be seen in Figure 3. A more detailed overview of task 2 can be found in Appendix D.3. During this task a total of nine multiplications were shown in a randomized order, which resulted in a duration of 225 seconds for task 2.

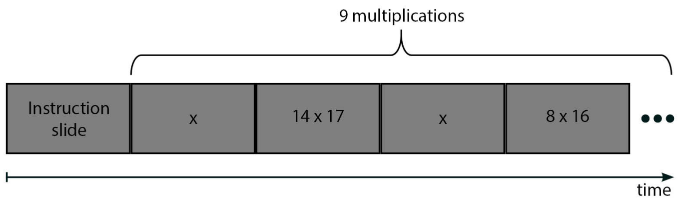


*Figure 3: Overview task 2*

#### 3.4.3 Task 3

Prior to the task, an instruction slide was shown to the participants [Appendix H.3]. Participants were instructed to follow a moving ball that appeared before them on the screen. Once the participant read and understood what was detailed on the instruction slide, he/she pressed the space bar and initiated the third task [see Figure 4]. Before every moving ball a control slide was shown for 10 seconds [Appendix E.3]. The duration of a single ball movement was 6 seconds. In total 24 different videos were shown which resulted in a duration of 384 seconds for task 3. A more detailed overview of task 3 can be found in Appendix D.4.

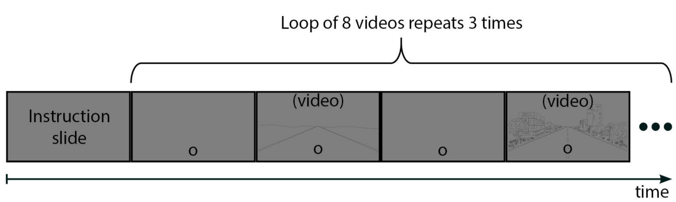


*Figure 4: Overview task 3*

### 3.5 Main MATLAB-script

Previous named tasks were implemented in the MATLAB-script ExperimentBEP [Appendix C.1]. In this script, all the controlslides, instructionslides, tones for task 1, multiplication slides and videos were loaded into MATLAB. Also the randomization of the multiplications and videos was programmed in this script. When everything was loaded in by ExperimentBEP, the script ExperimentBEPMainScript ran during the experiment [Appendix C.2]. This script used the loaded information from ExperimentBEP to run the experiment by sending the right frames to the monitor and the right pulses to the PowerRef III.

### 3.6 Data analysis

Both pulses send from the computer to the PowerRef III and the clock times of the computer were saved in MATLAB data-files. Furthermore, all data measured by the PowerRef III was automatically saved in CVS-files per participant. The PowerRef III measured the eyes 50 times per second, every row in this CSV-files represented one measurement. The columns showed the different variables the PowerRef III measured, see Appendix I for the list of the variables. A MATLAB-function readRaw [Appendix C.3] read 16 columns for further analysis, since not every variable was relevant. Out of this data, the useful variables were deducted for upcoming plots and answering the hypothesis.

The first time a stimuli was shown, it could cause a startle response, the third time the participant might be familiar with a certain objective. To filter these effects, all the tests were randomised. The right multiplications and videos were deducted from the data with the use of the pulses. By cutting the data off at the right pulses, the multiplications and videos were placed together easily. The script that makes these cuts and creates new datamatrices is called processRawData [Appendix C.4]. From these data sets, plots were generated and connections were made clear with the use of the MATLAB-script StatisticsFiltDataPaper [Appendix C.5].

With a paired-sample Student's t-test, any significant difference between two data sets can be proven. This was done by using the MATLAB-function ttest. Valuable output from this function gives the p-value, confidence interval (CI) and the degrees of freedom (df). The degrees of freedom in this case means the number of participants minus one. For this research, the p-value is set on 0.05. For any value below 0.05 there is a significant difference between the two data sets.

For the data-sets as input for the paired-sample Student's t-test, the following formula was used: The mean of the measurements during the control slide is taken and the maximum of the whole video. The difference between these two values was divided by the mean, this gives a fraction of the change between the baseline and the maximum value. This value times 100 gives a percentage.

# 4 Results

The 35 participants (20 males, mean age = 22.0 years; $SD = 1.59$) had an average refraction error of the right eye of -0.100 D and of the left eye of -0.071 D. Most participants had brown or blue eyes. 37.14% had brown eyes and 40.0% had blue eyes. 14.29% of the participants had green eyes. 5.72% had a mixture of brown and green coloured eyes and just one participant, 2.86%, had grey eyes.

The PowerRef III measured both eyes separately. For the data-analysis, the average of the left and right eye has been taken. This gave more clarity for the conclusions coming out the data-analysis, the number of graphs was halved because of this method.
As evidence that this could be done, the cross-correlation between the left and right eye was plotted for all tasks [Appendix J]. All cross-correlation graphs between the right and left eye showed a maximum at zero on the x-axis. This means that the graphs of the left and right eye are identical but translated over the y-axis. Just percentage differences were considered, so it is possible to conclude that this is a rightful way to take the average between the left and right eye.
The cross-correlation should not be confused with the auto-correlation, although the outcome would look alike. It was unknown if the two data-sets would be identical. When the identically of the data-sets would be known, then it would be an auto-correlation.

## 4.1 Task 1

Figure 5 shows two graphs. The light blue line represents the low tones, the dark blue line represents the high tones. The graph on top shows the refraction, measured in dioptre (D), plotted against the time (s). The bottom graph represents the gaze(X), shown in degrees (deg), measured over time (s).

A paired-sample Student's t-test -function t-test in MATLAB- was conducted for the refraction and the gaze in X direction for task 1, see table 3 and table 4. Both vectors used in the t-test are done at the level of participants and thus have the same length. This is the number of participants minus the participants who score outside the hard limits. In this test a difference has been made between the high and low tones.

To check the vergence, the gaze in x direction is plotted separate for both eyes with the high and low tones. The vergence is calculated as follows:

$$Vergence = |(Gaze(X)righteye) - (Gaze(X)lefteye)|$$

This is plotted for the left and right eye in Appendix L.1.

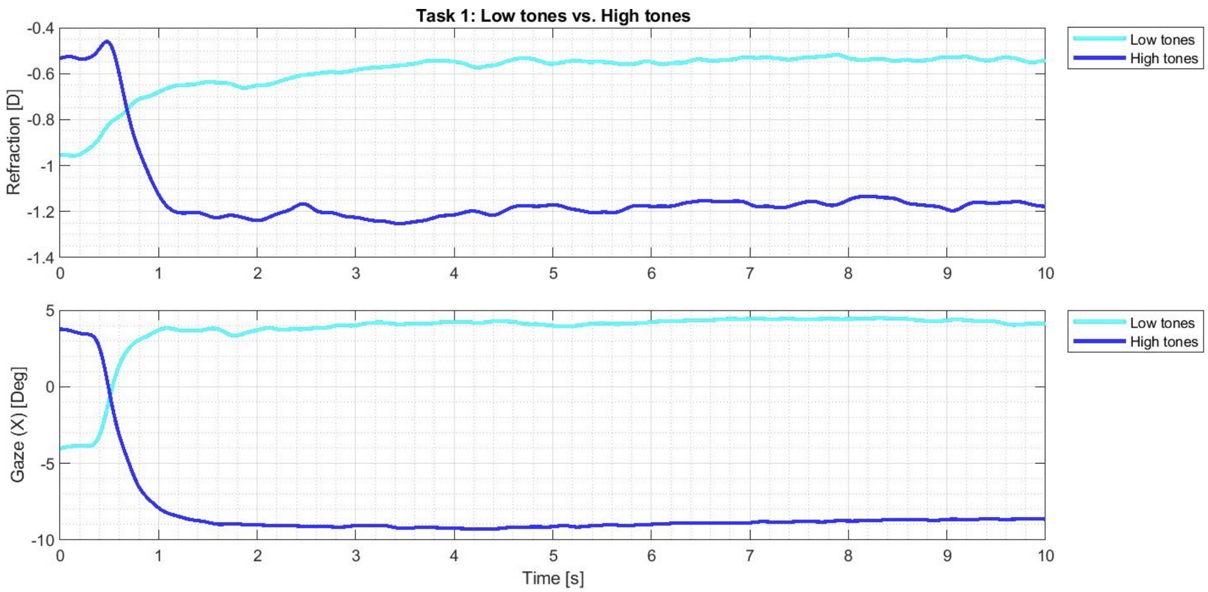


*Figure 5: Refraction and gaze(X) as a function of time for low and high tones. The low tones assisted the eyes to the far dot on the left. The high tones assisted the eyes to the dot nearby on the right.*

*Table 3: Paired-sample Student's t-test over two tones for Gaze(X); low and high. Direction left is positive and direction right is negative. The values for the mean and standard deviation (SD) are percentages if the difference between the high and low tones. 1 refers to low tones and 2 refers to high tones. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Frequency tone | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI (1) | CI (2) | df |
|---|---|---|---|---|---|---|---|---|
| Low - High | 3.70 | 1.07 | -8.17 | 1.74 | 4.880e-28 | 11.17 | 12.56 | 34 |

*Table 4: Paired-sample Student's t-test over two tones for the refraction; low and high. The values for the mean and standard deviation (SD) are percentages if the difference between the high and low tones. 1 refers to low tones and 2 refers to high tones. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Frequency tone | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI (1) | CI (2) | df |
|---|---|---|---|---|---|---|---|---|
| Low - High | -0.59 | 0.44 | -1.14 | 0.58 | 1.698e-12 | 0.45 | 0.66 | 34 |

## 4.2 Task 2

In task 2 three multiplications with three levels of difficulty were compared to each other. The easy multiplications are shown in green, the average multiplication are shown in blue and the difficult multiplications are shown in red. In Figure 6 the x-axis represents the time. The top figure shows the pupil diameter, in [mm], on the y-axis. The figure on the bottoms shows the refraction on the y-axis.

Scatter plots were made by taking the mean of a variable during the control slide and taking the maximum value during the shown stimuli (multiplications or videos). The percentage change was calculated by the formula:

$$((max - mean)/mean) * 100\%$$

This gave one percentage difference per participant during one task. When this was done for two variables that should be compared to each other, 35 X and Y values were composed in a figure and the scatter plots were made.

The pupil diameter and refraction are plotted against each other. For the easy, average and difficult multiplications scatter plots were made [Appendix K.1].

The three difficulties of the multiplications were tested with a paired-sample Student's t-test using the function ttest in MATLAB (table 5 and table 6). Both vectors used in the t-test are done at the level of participants and thus have the same length. This is the number of participants minus the participants who score outside the hard limits. Difference was made between the data-sets as easy, average and difficult multiplications.

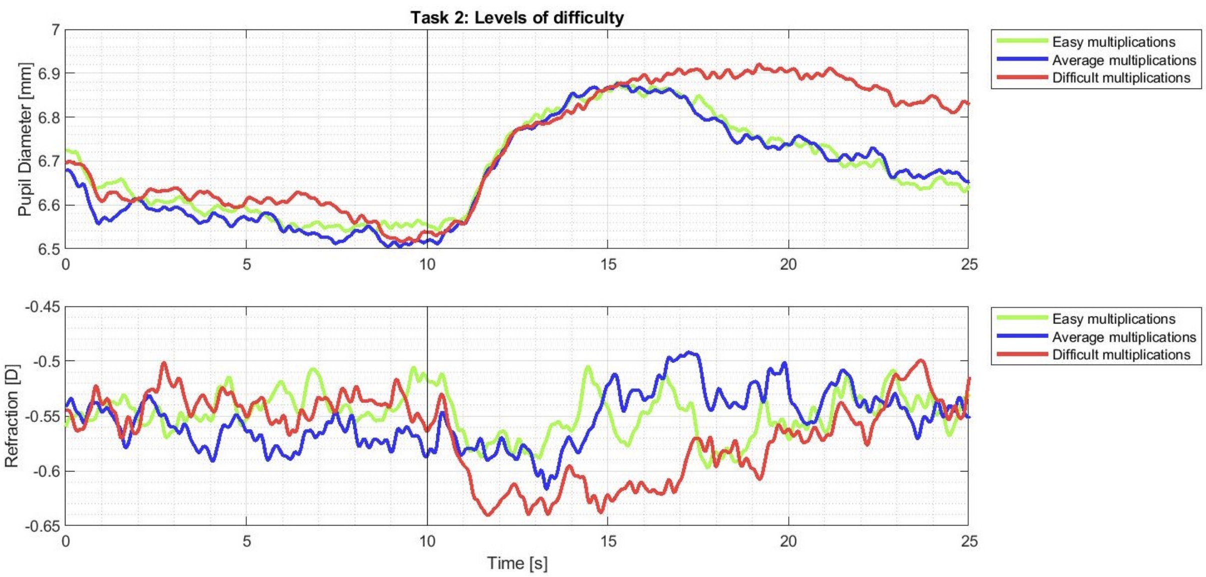


*Figure 6: Three levels of difficulty plotted alongside each other. Refraction and pupil diameter as a function of time. During the first ten seconds the control slide is shown. After these ten seconds the multiplications were shown.*

*Table 5: Paired-sample Student's t-test over three difficulties for the pupil diameter change; easy, average and difficult. The values for the mean are percentages of the average value of the control slide and the standard deviation (SD) is the difference of the mean with the maximum value of the whole multiplication set. 1 refers to the left column of 'difficulty multiplications' and 2 refers to the second column of 'difficulty multiplications'. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Difficulty multiplications | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| Easy - Average | -28.19 | 21.32 | -30.91 | 21.48 | 0.516 | -5.76 | 11.20 | 26 |
| Easy - Difficult | -26.25 | 23.32 | -17.57 | 40.67 | 0.162 | -21.06 | 3.71 | 27 |
| Average - Difficult | -36.57 | 27.61 | -31.28 | 38.68 | 0.252 | -14.55 | 3.96 | 29 |

*Table 6: Paired-sample Student's t-test over three difficulties for the refraction change; easy, average and difficult. The values for the mean and standard deviation (SD) are percentages of the average value of the control slide, and the difference with the maximum value of the whole multiplication set. 1 refers to the left column of 'difficulty multiplications' and 2 refers to the second column of 'difficulty multiplications'. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Difficulty multiplications | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| Easy - Average | 5.49 | 1.47 | 6.77 | 2.06 | 0.003 | -2.08 | -0.48 | 22 |
| Easy - Difficult | 5.46 | 1.52 | 6.50 | 2.14 | 0.040 | -2.03 | -0.05 | 24 |
| Average - Difficult | 6.66 | 2.08 | 6.46 | 2.27 | 0.718 | -0.95 | 1.36 | 23 |

## 4.3 Task 3

Task 3 distinguishes static and dynamic motion for the four different levels of realism of the roads. In Figure 7 the light blue line represents the static stimuli, the dark blue line shows the dynamic stimuli. The static and dynamic measurements are compared to each other in three separate graphs in Figure 7. Refraction, the pupil diameter and gaze(Y) were plotted against the time.

All Variables of the triad were plotted against each other for all static and dynamic stimuli in scatter plots: Pupil diameter against gaze(Y), pupil diameter against refraction and gaze(Y) against refraction. This was done for both static and dynamic stimuli. This results in six scatter plots [Appendix K.2.].

In Figure 8 the four levels are represented by different coloured lines. The refraction, pupil diameter and gaze(Y) are plotted against time. The green line stands for level 1, the blue line for level 2, the red line for level 3 and the pink line for level 4.

All variables of the triad were plotted against each other for all four levels in scatter plots: Pupil diameter against gaze(Y), pupil diameter against refraction and gaze(Y) against refraction on all four levels. This results in twelve scatter plots [Appendix K.2.].

The static and dynamic videos were tested with a paired-sample Student's t-test using the function ttest in MATLAB (table 7). Difference was made between the data-sets of static and dynamic videos. Both vectors used in the t-test are done at the level of participants and thus have the same length. This is the number of participants minus the participants who score outside the hard limits. The test was done for the variables of change in pupil diameter, refraction and gaze(Y).
Also for the different levels of realism, a paired-sample Student's t-test was done [see table 8, table 9 and table 10]. A difference between the four levels was made; no road, simple road, average road and realistic road. The variables were again the change in pupil diameter, refraction and gaze in y direction.

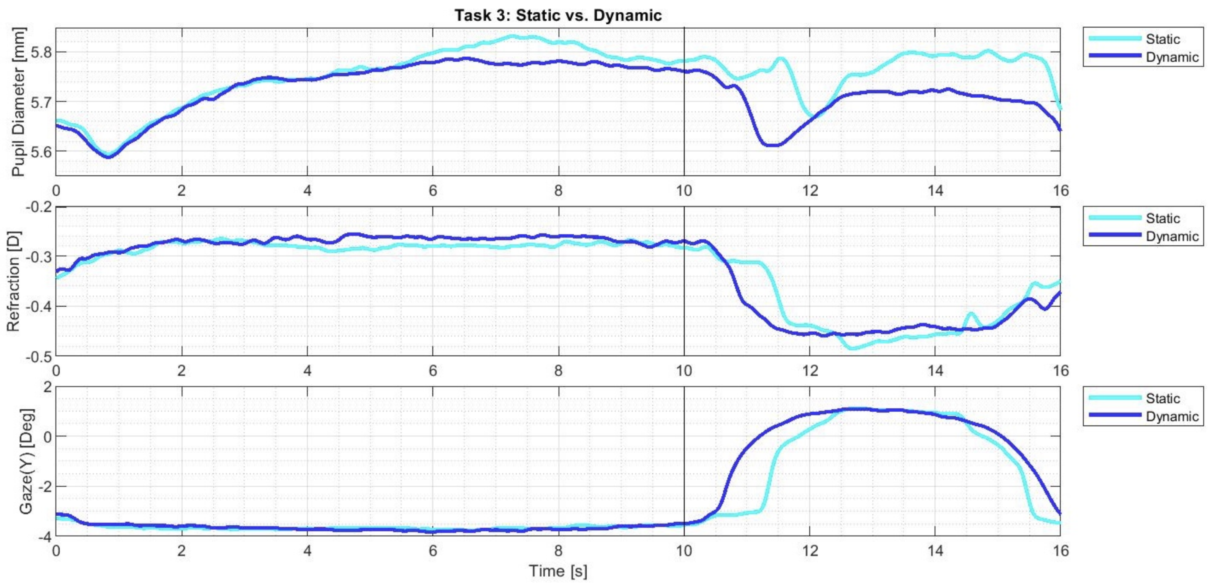


*Figure 7: Static stimuli plotted alongside dynamic stimuli. Pupil diameter, refraction and gaze(Y) were plotted against time. During the first ten seconds the control slide was shown. After these ten seconds the stimuli were shown.*

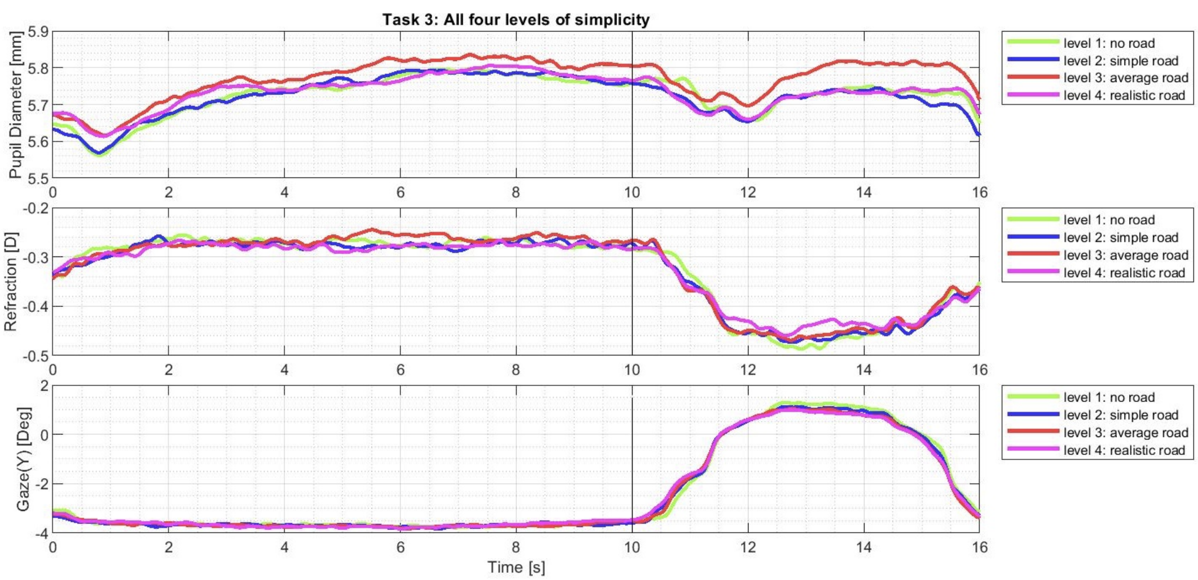


*Figure 8: The four levels of realism plotted alongside each other. Pupil diameter, refraction and gaze(Y) were plotted against time. During the first ten seconds the control slide was shown. After these ten seconds the stimuli were shown.*

*Table 7: Paired-sample Student's t-test over all static and dynamic videos, for the pupil diameter, gaze(Y) and refraction change. The values for the mean are percentages of the average value of the control slide and the standard deviation (SD) is the difference of the mean with the maximum value of the whole set of video's. 1 refers to the left column of 'type of video' (static) and 2 is refers to the second column of 'type of video' (Dynamic). Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Type of video | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| Static - Dynamic (Pupil diameter) | 2.36 | 1.44 | 2.90 | 1.20 | 0.097 | -1.17 | 0.11 | 23 |
| Static - Dynamic (Refraction) | -1.72 | 12.66 | -2.04 | 22.18 | 0.931 | -7.15 | 7.79 | 30 |
| Static - Dynamic (Gaze(Y)) | -129.38 | 61.97 | -126.60 | 59.72 | 0.324 | -8.46 | 2.89 | 30 |

*Table 8: Paired-sample Student's t-test over the different levels of realism for the pupil diameter change; no road, simple road, average road and realistic road. The values for the mean are percentages of the average value of the control slide and the standard deviation (SD) is the difference of the mean with the maximum value of the whole set of video's. 1 refers to the left column of 'levels of realism' and 2 refers to the second column of 'levels of realism'. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Levels of realism | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| No road - Simple road | 2.51 | 2.06 | 2.31 | 1.27 | 0.666 | -0.72 | 1.10 | 24 |
| No road - Average road | 2.57 | 1.95 | 2.40 | 1.09 | 0.671 | -0.62 | 0.94 | 22 |
| No road - Realistic road | 2.57 | 2.09 | 1.93 | 1.08 | 0.205 | -0.38 | 1.67 | 21 |
| Simple road - Average road | 2.13 | 1.28 | 2.41 | 1.10 | 0.362 | -0.92 | 0.35 | 18 |
| Simple road - Realistic road | 2.24 | 1.23 | 1.62 | 1.00 | 0.098 | -0.13 | 1.37 | 16 |
| Average road - Realistic road | 2.12 | 0.93 | 2.06 | 1.01 | 0.842 | -0.59 | 0.71 | 16 |

*Table 9: Paired-sample Student's t-test over the different levels of realism for the refraction change; no road, simple road, average road and realistic road. The values for the mean are percentages of the average value of the control slide and the standard deviation (SD) is the difference of the mean with the maximum value of the whole set of video's. 1 refers to the left column of 'levels of realism' and 2 refers to the second column of 'levels of realism'. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Levels of realism | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| No road - Simple road | -3.54 | 11.24 | -4.72 | 15.70 | 0.655 | -4.18 | 6.54 | 28 |
| No road - Average road | -3.54 | 11.24 | -2.03 | 14.02 | 0.589 | -7.14 | 4.13 | 28 |
| No road - Realistic road | -3.54 | 11.24 | 1.36 | 20.06 | 0.213 | -12.77 | 2.97 | 28 |
| Simple road - Average road | -4.89 | 17.30 | -1.77 | 17.84 | 0.198 | -7.97 | 1.72 | 30 |
| Simple road - Realistic road | -5.88 | 16.68 | 1.22 | 19.73 | 0.031 | -13.49 | -0.70 | 29 |
| Average road - Realistic road | -5.57 | 19.60 | 2.47 | 20.62 | 0.083 | -17.19 | 1.11 | 30 |

*Table 10: Paired-sample Student's t-test over the different levels of realism for the gaze(Y) change; no road, simple road, average road and realistic road. The values for the mean are percentages of the average value of the control slide and the standard deviation (SD) is the difference of the mean with the maximum value of the whole set of video's. 1 refers to the left column of 'levels of realism' and 2 refers to the second column of 'levels of realism'. Other values displayed are the p-value, Confidence Interval (CI), and the degrees of freedom (df).*

| Levels of realism | Mean 1 | SD 1 | Mean 2 | SD 2 | p-value | CI(1) | CI(2) | df |
|---|---|---|---|---|---|---|---|---|
| No road - Simple road | -142.11 | 70.55 | -138.34 | 66.73 | 0.332 | -11.57 | 4.03 | 31 |
| No road - Average road | -142.11 | 70.55 | -134.05 | 72.90 | 0.090 | -17.46 | 1.33 | 31 |
| No road - Realistic road | -142.11 | 70.55 | -133.81 | 76.54 | 0.093 | -18.07 | 1.46 | 31 |
| Simple road - Average road | -138.34 | 66.73 | -134.05 | 72.90 | 0.359 | -13.69 | 5.11 | 31 |
| Simple road - Realistic road | -138.34 | 66.73 | -133.81 | 76.54 | 0.343 | -14.13 | 5.07 | 31 |
| Average road - Realistic road | -134.05 | 72.90 | -133.81 | 76.54 | 0.953 | -8.47 | 7.99 | 31 |

# 5 Discussion

The aim of this research was to investigate whether there is a correlation between the pupil size, the accommodation and the vergence of the eye.

## 5.1 Task 1

During the first part of the study, task 1 in the experiment, accommodation was measured with 40 centimeters between the points of fixation. Figure 5, shown in the results, clearly shows more refraction of the lens, so more dioptres, when the high tone was presented. The high tone was equivalent to the stick closer to the eye. The left stick was placed twice as far as the right stick. Table 4 shows that the mean of the refraction for the low tones -this corresponds with the left stick- is -0.58 D. The mean of the refraction for the high tones is -1.14 D. For the stick that is twice as far away from the eyes (high tones), the mean refraction is half of the mean refraction of the stick nearby (- 0.58 * 2 = - 1.16). The small aberration is probably due to inaccuracies in the measurements and measuring device.

The graph in Figure 5 where gaze(X) is plotted over time verifies that the participants fixated their eyes on the correct sticks.

**Paired-sample Student's t-test:** To prove that both low and high tone gave different values for refraction and the gaze in x direction, a paired-sample Student's t-test was conducted, see table 3 and table 4. For both variables the p-value is smaller than 0.05, which indicates that there is a significant difference between the low and high tones for the refraction.

**Vergence calculation:** In Appendix L.1 the vergence is calculated for both the left and right eye. When calculating these values by hand -the dots are horizontally 14.0 cm away from each other- A vergence of 4.27° for the low tones should have occurred and 8.33° for the high tones. A vergence approximately twice as big for the high tones was predicted since the distance is doubled. The graph for the vergence in Appendix L.1 for the low tones stabilizes around 5°, which is acceptable, but the high tones converge to 1.5°. This is not in line with the expectation, the reason why this occurred is unclear. The expectation would be an increase in vergence when looking at a closer object.

## 5.2 Task 2

During the second task the pupil diameter and the refraction were looked at while performing 9 multiplications, divided into three levels of difficulty. In Figure 6 these three levels of difficulty are shown in two graphs. Considering all multiplications, the result is in line with the literature. The three lines are taken together because there is no significant difference between the three levels of difficulty. This is made clear with the p-value in table 5. **Pupil diameter:** The pupil dilates when performing cognitive tasks. Participants gave an answer on the easy and average multiplication within 15 seconds. This is visible in the graph, the pupil constricts after the answer is given. Most participants could not answer the difficult multiplications in time, this explains why the pupil does not constrict again to its baseline.

The easy and average multiplication show comparable results for the pupil diameter, this might be due to the fact that the levels of the multiplications were not far enough apart. It is also questionable which multiplication is experienced as hard for the different participants.

**Refraction:** The refraction is shown in the second subplot of Figure 6. There is not a clearly defined pattern to the graph, although it seems like the refraction of the lens is a bigger value for the set of hard multiplications.

**Paired-sample Student's t-test:** The p-values in table 6 show that the multiplication sets Easy - Average and Easy - Difficult are significantly different (p-value smaller than 0.05).

Unfortunately, this is not the case for the set Average - Difficult. It seems that the difference between Average and Difficult is not big enough for the refraction.

**Filtering outliers t-test/ scatterplot:** The degrees of freedom are less than 34 (number of participants minus one), since there were hard limits encoded in the t-test, which filtered outliers. Also in the scatter plots the hard limits were used to filter outliers. This is done so the outliers did not skew the whole test/graph.
Filtering the outliers was done as follows: for each variable (on the x- and y-axis) the mean of all 35 points was calculated, as well as the standard deviation. Simply every value grater than the mean plus the standard deviation -or every value below the mean minus- the standard deviation was filtered out. These edge-values are the hard limits. This is the reason that not every t-test and scatter plot contains the same amount of participants.

**Sub-question 1:** The first sub-question was: Does mental effort (performing multiplications) result in a change in accommodation or an increase in pupil diameter?
A hypotheses was formed, the expectation was that only the pupil size would change as a result of performing mental effort, the accommodation would not change. The findings are in line with the hypotheses. The pupil dilated when doing mental effort and constricted after the effort was done. The accommodation does not change by doing mental effort.

## 5.3 Task 3

During the third task, static and dynamic stimuli were compared to each other and four levels of realism were considered. In figure 7 the static and dynamic stimuli are plotted. No clear differences can be detected between static and dynamic. This is substantiated by the p-values in table 7.

### 5.3.1 Static and dynamic

Starting with the gaze(Y), the same motion is visible [figure 7]. The eyes follow the ball in steps when looking at static motion instead of a fluent movement. Exactly the same result is seen when looking at the refraction plotted over time. It is clearly visible in the graph of the gaze(Y) that there are steps from the static stimuli. This should be four steps, since the change between the static positions is equal to four. This can be seen at 11.5, 12.5, 14,5 and 15.5 seconds. From this information we see a reaction time of the eye of +/- 0.5 seconds. There is a small difference in pupil diameter between the static and dynamic stimuli. The pupil diameter is a little bit smaller for the dynamic stimuli, but this is not significant, as shown in table 7. For all three, the means of the static and dynamic videos are very close together, for all variables. This is also supported with the p-values.

**Sub-question 2:** A conclusion can be drawn for the second sub-question: Does it make a significant difference to the amount of accommodation, gaze shift or change in pupil size if the movement on a 2D screen is static instead of dynamic?
It does not make a significant difference for the amount of accommodation, gaze shift or change in pupil diameter if the movement in a 2D screen is static instead of dynamic.

### 5.3.2 Four levels of realism

**Refraction change:** The expectation was that more accommodation would occur, when more depth cues were present. This means that expected was that the biggest differences would be seen between level 1 (no road) and level 4 (realistic road). The accommodation would change the most in level 4 and the least in level 1. In table 9, these two are compared to each other. The p-value is not significant, there is not a significant difference between the refraction change of level 1 and level 4. Most p-values stay above the value of 0.05. There is one exception for the refraction change between the simple road and the realistic road. Here is the p-value smaller than 0.05. Since this is the only case, no general assumption about any significant difference between the levels can be made. Furthermore does the change between the means stay almost constant.

**Gaze shift(Y):** For the gaze shift(Y) a hypothesis was formed that the magnitude of the gaze shift(Y) would not change as a result of more depth cues. In figure 8 the four levels all show the same result for the gaze in y direction. The gaze(Y) increases when the ball moves up -so in the first three seconds of the video- and decreases when the ball moves downwards during the last three seconds of the video. In table 10, the p-value is above 0.05 for all t-tests and mean 1 and mean 2 are very close together. This means that that the depth cues do not influence the magnitude of the change in gaze(Y). This is in line with the hypothesis.

The values for the mean and standard deviation of the change in gaze(Y) are extremely negative, since in the equation for the paired-sample Student's t-test the control slide was implemented as well. This control slide made participants focus on a point under a relatively negative angle in the y direction for 10 seconds (-3.8°). When the video starts, these values become positive (maximum 1°). However, since the video lasted only six seconds, the mean always stayed negative. The range could have been taken differently, or absolute values could have been used for the gaze(Y).

**Pupil diameter:** Looking at the pupil diameter 8, it does stabilize for every level to a certain value at the end of the control slide. Then when the video starts, the pupil constricts for the first seconds and stabilizes again before it constricts just before the end of the video. This behaviour is similar for all levels [see p-value in table 8]. A possible conclusion for the first constriction of the pupil is that the pupil diameter does not change during the videos due to the illusions, but it is due to a startle response. The control slide only consisted of the ball, at the beginning of the video surroundings popped up within a frame, this caused the startle response. The constriction at the end of the video is due to the pupillary near response. The eyes converge and the pupil constricts.

The hypotheses formed for the pupil size was that the pupil diameter does not depend on the amount of depth cues in a stimuli. The findings are in line with this hypotheses. All levels show the same shaped line. In table 8 the p-values indicate that there are no significant differences between different levels of realism (all p-values are larger than 0.05). Also, the means of the different levels tested against each other are quite alike. All this indicates that the pupil size does not depend on the amount of depth cues.

**Paired-sample Student's t-test:** Looking at figure 8, where the different levels of realism are shown, not much differences can be seen between the levels, this is also visible in table 8, table 9 and table 10. All p-values are above 0.05, so there are no significant differences between different levels of realism. The depth cues do not cause a different response of the eye. This result is not in line with the hypotheses.

**Sub-question 3:** The last sub-question is answered; Does the number of depth cues that a static or dynamic 2D-stimuli contains, makes a difference in the way accommodation, the gaze shift and pupil size change?

The number of depth cues a static or dynamic 2D-stimuli contains, does not make a difference in the change of accommodation, gaze shift(Y) and pupil size. The different levels can be reviewed as one.

Prior research -as confirmed with task 1- showed that near object cause a more negative value for the refraction, thus an increase in accommodation [figure 5]. Expectations were

that because of the illusion of the ball travelling farther away from the participant, the refraction would becomes less negative as well. However, the refraction becomes more negative when the ball moves away from the participant, as can be seen in figure 8. The refraction becomes less negative when the ball is coming closer. This is happening for all levels and is against measurements of task 1. The reason for this horizontal inverted behaviour is unclear.
It has to be taken into account that these values were measured with infra-red rays, which refract on the surface of the eye. Since the eyes surface is not completely smooth -caused by little bumps, bacteria, etcetera- it is not sure if the refraction the PowerRef III has measured from the eyes is completely reliable.
It could also be possible that it had something to do with the orientation of the eye. The PowerRef III measured straight from the eye, but it is possible that when the eyes of the participant made a rotation (follow the ball), the bulging of the eye changed at the place where the rays entered the eye. It could be possible that the more rotation the eyes made, the more change in refraction was measured due to this difference in bulging. For now, this is a possible reason why the refraction had an opposite shape as expected.

## 5.4 Conclusion

In sum, no correlation was found between any of the three variables of the triad during mental activity and looking at 2D images. The pupil size, the accommodation and the vergence did not depend on each other. These findings are consistent with the hypothesis of the research question.
Prior studies that made use of pupillometry on a 2D screen will not be judged on the fact that they did not take accommodation and vergence into account while measuring the pupil size.

## 5.5 Recommendations

Several things in this study could have influenced the measurements. For this reason a p-value of 0.05 was chosen, in order not to dismiss correlations falsely. In future research a p-value of 0.005 could be chosen [1], this would be more robust.
In the first instance a LED light box was used as lighting during the experiment. With most participants the pupil size became to big or to small for the PowerRef III to measure the eyes. The range of the pupil diameter that the PowerRef III can measure, is between the 4.0 mm and 8.0 mm. The measurements where the LED light was used were not taken into account. The cause of these wrong measurements is not clear.
Instead of the LED light box, a desk lamp was used during the experiments. With light coming from the LED light box the pupil size became too big with some participants. Also, the illumination could not be kept constant. This was because every time the LED light box would turn on and off, the light intensity would change as well. The power button was able to twist to change the light intensity. Measurements taken from these participants were excluded from the data used in this research. The light intensity was measured with an illuminance meter (T-10A, Konica Minolta, Inc.). Measurement were taken with the illuminance meter right where the participants eyes would be, with the sensor directed to the computer screen. Values measured were; 9.8 lx with the LED light box and 1.15 lx with the desk lamp.

Whenever the pupil diameter became to big to measure during the experiment, the PowerRef III interpolated between the last value that could be measured and the first value after no measurements. This caused incorrect results over small periods of time. In further research, these interpolated results should be filtered out. It can not be made sure what the measured values were in that interval. It is even better if in future research the PowerRef

III, or another eye-measuring device, is able to measure a broader range of the pupil size.

The scatter plots could have been more meaningful. Gaze(Y) changes of -340% (Figure K.9) seems strange. In future research maybe some plots could better be displayed in concrete values instead of percentages.

# A Appendix A

## Depth cues

Occlusion
Occlusion appears when near objects overlap objects that are more in the distance. The object in the front is fully visible, the view of the one behind is partially blocked.

Relative height
The cue of relative height means that objects that are more in the distance are placed higher in an image. This is true for objects underneath the horizon. For objects above the horizon; they appear to be closer the higher they are placed above the horizon. As in figure A.1, all three men are the same size, but the higher they stand in the image, the further they seem to be away. The clouds above the horizon appear to be closer, the higher float.

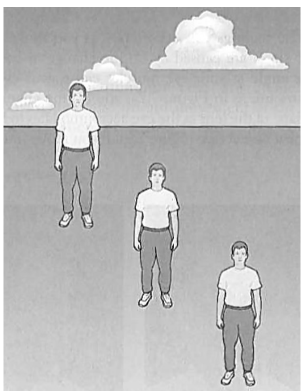

*Figure A.1: Example of relative height; The higher the man is placed in the image, the further away he seems to be [6].*

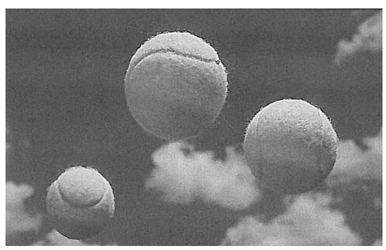

*Figure A.2: Example of relative size; The smaller the same ball, the further away it seems to be [6].*

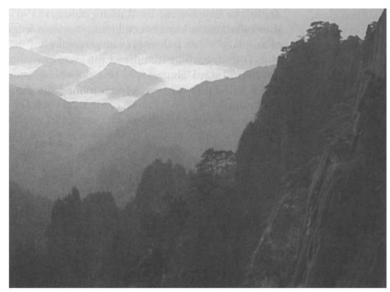

*Figure A.3: Example of atmospheric perspective; The hazier the part of the image, the further away it seems to be [6].*

Cast shadow
The place of a shadow provides information about the location of an object.

Relative size
The relative size of objects gives insight into the relative depth of objects. If it is clear that several objects have the same size, relative size cues give insight in on the relative depth of the objects. If several three tennis balls are shown, the size they have compared to each other, shows how far they are away. The smaller the three, the further away it seems to be, this is shown in figure A.2.

Familiar size
Prior knowledge of actual sizes influences the perception of depth. The projected size can be compared to information from previous seen objects, to determine the depth of objects.

Atmospheric perspective

Objects nearby are more sharp. Distant objects are more hazy. This is due to the air that has different small things in it, like water, dust and pollutants. So the hazier an object is, the further it appears to be. An example of this depth cue is shown in figure A.3.

Linear perspective

The depth cue of linear perspective refers to the parallel lines that converge in the distance. The more the lines converge, the greater the distance is. At infinity these lines meet in one point.

Texture gradient

Goldstein (2002) describes texture gradient as follows: ”Elements that are equally spaced in a scene appear to be more closely packed as distance increases, such as square tiles in the floor”. So for example with the stripes that separate two lanes on a road, the more the stripes are in the distance, the closer they are together. The cue of texture gradient is usually located on the floor. The perception of depth is less when the floor is removed.

Motion parallax

When sitting in a train, nearby objects move away fast, distant objects seem to move slower. Motion parallax says something about about the speed of movement for far and near objects. Objects far away move slow, near objects speed

Deletion and accretion

There are two surfaces, when an observer moves, the surfaces overlap or one gets covered up by the otherone. The way they move relative to each other gives insight into their place.

Depth from motion

This depth cue only applies to moving objects. An object that moves away from an observer, seems to move away when the size of the objects gets smaller. This change in size gives a perception of depth. When the objects moves towards the eyes, it gets bigger, this gives the perception that the objects comes closer to the eye [6], [5].

# B Appendix B

## B.1 Inform consent

See next page.

# Consent form for participants

Research Title: “Accommodation As A Possible Confounder In Pupillometrics Research”

**Researchers:**
Tamir Themans – Researcher — Email: T.S.Themans@student.tudelft.nl
Stefan Jansen – Researcher — Email: S.T.Jansen@student.tudelft.nl
Julia Russell – Researcher — Email: J.N.M.Russell@student.tudelft.nl

Ir. Lars Kooijman – Supervisor — Email: l.kooijman-1@tudelft.nl
Dr.ir. Bastiaan Petermeijer – Supervisor — Email: s.m.petermeijer@tudelft.nl
Dr. Dimitra Dodou – Supervisor — Email: d.dodou@tudelft.nl
Dr.ir. Joost de Winter – Supervisor — Email: j.c.f.dewinter@tudelft.nl

**Location of the experiment:**
Room 34 A-0-811 (above lecture room C), see Figure 1.
Faculty of Mechanical, Maritime and Materials Engineering, Delft University of Technology, Mekelweg 2, 2628 CD Delft


*Figure 1: Location of the experiment (door at the right side, then up the stairs to the right)*

**Introduction:** Please read this consent document carefully before you decide to participate. This document describes the purpose, procedures, and potential risks/discomforts. Your signature is required for participation.

If you have any form of eyesight deficiency (i.e., near- or far-sightness) and you wear contact lenses, please wear these during the experiment. The measurement equipment functions better with contact lenses than glasses, so please try to wear contact lenses instead of glasses. Due to measurement errors, glasses will not be accepted when participating in the experiment.

**Purpose of the study:** The aim is to investigate whether accommodation, pupil size, and vergence are influenced by mental workload and illusion of depth.

**Duration:** Your participation in this experiment will last approximately 15 minutes.

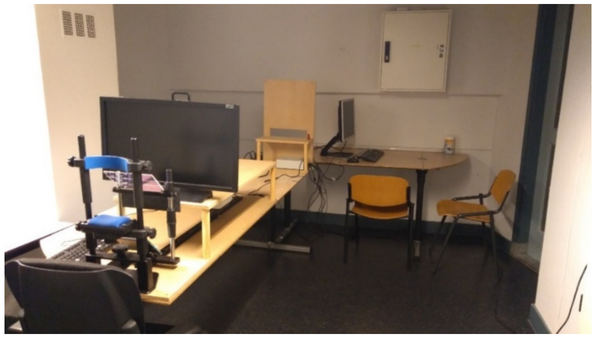

*Figure 2: Experimental setup with head support and eye tracker*

## Procedures and instructions

**Before the experiment starts**: You will be asked to complete a short questionnaire about your gender, age, refractive error (please check on beforehand when your last eye control has been done), and eye colour. You will then be asked to place your head in the head support shown in Figure 2.

**During the experiment**: Your first task will be to visually focus at the marked ends of two sticks placed in front of you. Your second task will be to mentally solve a number of multiplications that are shown to you on a computer screen and to audibly state your answer. Your third task will be to follow a ball shown on the screen.

**After the experiment**: You can ask the experimenter about your performance after the completion of the entire experiment.

**Risks and discomforts:** Some multiplications may be experienced as difficult, and due to the time constraint, you may experience some frustration. However, there are no known risks for you in this study. Some minor eyestrain or discomfort may arise from looking at the screen. If at any point you begin to feel uneasy for any reason, please do not hesitate to inform the experimenter so that you take a break to counteract any such symptoms.

**Anonymity:** All data collected in this study will be stored anonymously. You will not be personally identifiable in any future publications based on this work or in any data shared with other researchers.

**Right to refuse or withdraw:** Your participation in this study is entirely voluntary. You have the right to refuse or withdraw from this experiment at any time, without any negative consequences, and without needing to provide any explanation.

**Questions:** For any questions, you can contact one of the researchers at the email addresses provided above.

I have read and understood the information provided above. I give permission to store and use of collected data for the purposes of this study described above. The results of the study will not be made available in a way that could reveal the identity of individuals. I voluntarily agree to participate in this study.

Name:

……………………………………………………………………………………………..

Signature:

……………………………………………………………………………………………..

Date:

……… / …….. / ………

## B.2 Questionnaire

See next page.

# Questionnaire

Research Title: *'Accommodation As A Possible Confounder In Pupillometrics Research'*

**Researchers:**
Tamir Themans – Researcher Email: T.S.Themans@student.tudelft.nl
Stefan Jansen – Researcher Email: S.T.Jansen@student.tudelft.nl
Julia Russell – Researcher Email: J.N.M.Russell@student.tudelft.nl

Ir. Lars Kooijman – Supervisor Email: l.kooijman-1@tudelft.nl
Dr.ir. Bastiaan Petermeijer – Supervisor Email: s.m.petermeijer@tudelft.nl
Dr. Dimitra Dodou – Supervisor Email: d.dodou@tudelft.nl
Dr.ir. Joost de Winter – Supervisor Email: j.c.f.dewinter@tudelft.nl

1. Gender:
   - Male
   - Female
   - Other

2. Age: ______________

3. Eye colour: ___________________________________________________________

4. Do you have a refraction error?
   - No
   - Yes

4a. If filled in Yes, what is your refraction error?

Left eye: ___________________________

Right eye: ___________________________

4b. When were these values measured? (month/year) ______________

4c. Are you wearing contact lenses right now?

- No
- Yes

Thank you for filling in this form and participating in the experiment!

# C Appendix C

## C.1 ExperimentBEP.m

See next page.

```
function varargout = ExperimentBEP(varargin)
% EXPERIMENTBEP MATLAB code for ExperimentBEP.fig
%      EXPERIMENTBEP, by itself, creates a new EXPERIMENTBEP or raises the existing
%      singleton*.
%
%      H = EXPERIMENTBEP returns the handle to a new EXPERIMENTBEP or the handle to
%      the existing singleton*.
%
%      EXPERIMENTBEP('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in EXPERIMENTBEP.M with the given input arguments.
%
%      EXPERIMENTBEP('Property','Value',...) creates a new EXPERIMENTBEP or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ExperimentBEP_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ExperimentBEP_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ExperimentBEP

% Last Modified by GUIDE v2.5 30-Apr-2019 10:47:00

% Begin initialization code - DO NOT EDIT
    global super
    gui_Singleton = 1;
    gui_State = struct('gui_Name',       mfilename, ...
                       'gui_Singleton',  gui_Singleton, ...
                       'gui_OpeningFcn', @ExperimentBEP_OpeningFcn, ...
                       'gui_OutputFcn',  @ExperimentBEP_OutputFcn, ...
                       'gui_LayoutFcn',  [] , ...
                       'gui_Callback',   []);
    if nargin && ischar(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargout
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end
end
% End initialization code - DO NOT EDIT

% --- Executes just before ExperimentBEP is made visible.
function ExperimentBEP_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ExperimentBEP (see VARARGIN)
    global super
% Choose default command line output for ExperimentBEP
    handles.output = hObject;
% Update handles structure
    guidata(hObject, handles);
end

% UIWAIT makes ExperimentBEP wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = ExperimentBEP_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
```

```
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
    global super
    %%%%%%%%%%%%%%%
    rng('shuffle'); % MAKES SURE RANDOM IS REALLY RANDOM!!!
    %%%%%%%%%%%%%%%
    opengl hardware
    varargout{1}                 = handles.output;
    % Pre-allocate space experiment 1
    enter                        = 0;
    ClockSoundLow                = zeros(3,6);
    ClockSoundHigh               = zeros(3,6);
    elapsedTimeLow1              = zeros(3,1);
    elapsedTimeHigh1             = zeros(3,1);
    ClockStartExperiment1        = zeros(1,6);

    % Pre-allocate space experiment 2
    ClockCross                   = zeros(9,6);
    ClockProblem                 = zeros(9,6);
    reactionTime                 = zeros(9,1);
    elapsedTime2                 = zeros(9,1);
    ClockStartExperiment2        = zeros(1,6);

    % Pre-allocate space experiment 3
    ClockControlBall             = zeros(24,6);
    ClockMoveBall                = zeros(24,6);
    elapsedTime3                 = zeros(24,1);
    ClockStartExperiment3        = zeros(1,6);

%%  Organize Instructions
%   Find and organise instruction slides
    handles.InstructionMap       = super.InstructionMap;                   %
Stimuli background folder location
    fileInstruction              = fullfile(handles.InstructionMap,...
                                                      '*.jpg');          % Find
all the images in the folder
    allInstructions              = dir(fileInstruction);                   % Find
all the images in the folder
    handles.InstructionFileNames = {allInstructions.name};                 % Names
of all the images
    handles.numberOfInstructions = length(handles.InstructionFileNames);   % Find
the number of stimulus images in the folder

%%  Organize Control Slides
%   Find and organise control slides
    handles.ControlMap           = super.ControlMap;                       %
Stimuli background folder location
    fileControl                  = fullfile(handles.ControlMap, '*.jpg');  % Find
all the images in the folder
    allControl                   = dir(fileControl);                       % Find
all the images in the folder
    handles.ControlFileNames     = {allControl.name};                      % Names
of all the images
    handles.numberOfControl      = length(handles.ControlFileNames);       % Find
the number of stimulus images in the folder

%%  Organize Sounds
    Fs                  = 44100;
    time               = 0:1/Fs:1-1/Fs;
    Amplitude_low       = 15;
    Amplitude_high      = 5;
    Frequency_low       = 100;
    Frequency_high      = 1000;
    Pause               = 10;
    Number_of_sequences = 3;
```

```
    tone_low            = Amplitude_low*sin(2*pi*Frequency_low*time);      % Lage
toon
    tone_high           = Amplitude_high*sin(2*pi*Frequency_high*time);    % Hoge
toon
    player_low          = audioplayer(tone_low, Fs);
    player_high         = audioplayer(tone_high, Fs);

%%  Organize Multiplications
%   Find and organise multiplication slides
    handles.MultiplicationMap       = super.MultiplicationMap;             %
Stimuli background folder location
    fileMultiplication              = fullfile(handles.MultiplicationMap,...
                                               '*.jpg');                   % Find
all the images in the folder
    allMultiplications              = dir(fileMultiplication);             % Find
all the images in the folder
    handles.MultiplicationFileNames = {allMultiplications.name};           % Names
of all the images
    handles.numberOfMultiplications = length(handles.MultiplicationFileNames);%
Find the number of stimulus images in the folder
    handles.randomOrderMultiplications =
randperm(handles.numberOfMultiplications)    % Create an array for the random
presentation of the images
    super.RandomOrderMultiplications   = handles.randomOrderMultiplications;
%%  Organize Videos
%   Find and organise videos
    handles.VideoMap                 = super.VideoMap;                     %
Stimuli background folder location
    VideoMap                         = handles.VideoMap;
    rootFolder                       = pwd;
    cd(VideoMap)
    tic
    load('AllVideos.mat');
    toc
    T = 1/VideoCell{3,1}:1/VideoCell{3,1}:181/VideoCell{3,1};
    cd(rootFolder);
    handles.VideoFileNames           = VideoCell{1,1};                     % Names
of all the images
    handles.numberOfVideo            = length(handles.VideoFileNames);     % Find
the number of stimulus images in the folder
    handles.randomOrderVideo         = [randperm(handles.numberOfVideo);...
                                        randperm(handles.numberOfVideo);...
                                        randperm(handles.numberOfVideo)]   % Create
an array for the random presentation of the images
    super.RandomOrderVideos          = handles.randomOrderVideo;
    AllVideoFrames                   = VideoCell{2,1};                     % Locate
all video frames
    super.RandomOrderVideos          = handles.randomOrderVideo;
%% -------------------------Start Instruction 1-------------------------%%
    img_Instruction        =
imread([handles.InstructionMap,char(handles.InstructionFileNames(1))]);      %
Preload the control image
    imshow(img_Instruction);
    getkeywait(9000);
    ClockStartExperiment1(1,:)  = clock;
    pulseStartExperiment1       = 's10';
    fprintf(super.s,pulseStartExperiment1);                               % Pulse
indicating start control trial

%% -------------------------Start Sounds---------------------------------%%
    %Which Pulses are we going to log?
    %Which clocks are we going to log?

    ControlSlideExperiment1   = handles.ControlFileNames{1};               % Create
path for the selected image
    img_control               =
imread([handles.ControlMap,ControlSlideExperiment1]); % Preload the control slide
    imshow(img_control);
```

```
    for k = 1:Number_of_sequences
        pulseStartSoundLow        =
strcat('s0',num2str(k));                    % Compute pulse number start trial.
Pulse number corroborates with multiplication number
        ClockSoundLow(k,:)        = clock;                              % Store
computer time of control slide
        tic
        fprintf(super.s,pulseStartSoundLow);                            % Pulse
indicating start control trial
        playblocking(player_low);
        elapsedTimeLow1(k,1)   = toc;                                   % Store
stimulus time
        pause(Pause);

        pulseStartSoundHigh       =
strcat('s9',num2str(k));                   % Compute pulse number start trial.
Pulse number corroborates with multiplication number
        ClockSoundHigh(k,:)       = clock;                              % Store
computer time of control slide
        tic
        fprintf(super.s,pulseStartSoundHigh);                            % Pulse
indicating start control trial
        playblocking(player_high);
        elapsedTimeHigh1(k,1)   = toc;                                  % Store
stimulus time
        pause(Pause);
    end
    pulseEndExperiment1       = 's20';
    ClockEndExperiment1       = clock;

    fprintf(super.s,pulseEndExperiment1);                               % Pulse
indicating start control trial

    super.ClockSoundLow_Experiment1   = ClockSoundLow;                  % Write
control slide times in super
    super.ClockSoundHigh_Experiment1  = ClockSoundHigh;                 % Write
control slide times in super
    super.elapsedTimeLow_Experiment1  = elapsedTimeLow1;                % Write
trial times in super
    super.elapsedTimeHigh_Experiment1 = elapsedTimeHigh1;               % Write
trial times in super
    super.StartExperiment1            = ClockStartExperiment1;
    super.EndExperiment1              = ClockEndExperiment1;

%% -------------------------Remove Object--------------------------------%%
    img_Instruction       =
imread([handles.InstructionMap,char(handles.InstructionFileNames(4))]);
    imshow(img_Instruction);

    while enter ~= 13
        enter = getkeywait(900);
    end
    enter = 0;

%% -------------------------Start Instruction 2-------------------------%%
    img_Instruction       =
imread([handles.InstructionMap,char(handles.InstructionFileNames(2))]);
    imshow(img_Instruction);
    getkeywait(9000);
    ClockStartExperiment2(1,:)  = clock;
    pulseStartExperiment2       = 's30';
    fprintf(super.s,pulseStartExperiment2);                             % Pulse
indicating start control trial

%% -------------------------Start Multiplications------------------------%%

    ControlSlideExperiment2   = handles.ControlFileNames{2};
```

```
    img_control               =
imread([handles.ControlMap,ControlSlideExperiment2]); % Preload the control slide

    for k = 1 : handles.numberOfMultiplications
        filenumber         =
handles.randomOrderMultiplications(k);                        % Select the image
number from the random array
        fullFileName       = handles.MultiplicationFileNames{filenumber};  % Create
path for the selected image
        MultiplicationMap  = handles.MultiplicationMap;
        pulseStart         =
strcat('s7',num2str(filenumber));                       % Compute pulse number start
trial. Pulse number corroborates with multiplication number
        pulseReaction      = strcat('s8',num2str(filenumber));              %
Compute pulse number reaction time. Pulse number corroborates with 10 +
multiplication number
        pulseEnd           = strcat('s9',num2str(filenumber));              %
Compute pulse number end trial. Pulse number corroborates with 20 + multiplication
number
        img                = imread([MultiplicationMap,fullFileName]);     %
Preload the random stimulus image
        fprintf('Multiplication :%s \n',fullFileName)
%       Show Controlslide
        imshow(img_control);
        axis off;
        ClockCross(k,:)  = clock;                                          % Store
computer time of control slide
        fprintf(super.s,pulseStart);                                       % Pulse
indicating start control trial
        pause(10);                                                         %
Showing control slide for 10 seconds

%       Show The Randomly Selected Image
        imshow(img);
        axis off;
        ClockProblem(k,:)  = clock;                                        % Store
computer time of stimulus image
        tic                                                                % Start
stimulus
        [~, rt]            = getkeywait(15);                               % Wait
15 second or until a key is pressed and return key and reaction time
        fprintf(super.s,pulseReaction);                                    % Pulse
indicating reaction time
        pause(15-rt);                                                      % Wait
remaining time
        fprintf(super.s,pulseEnd);                                         % Pulse
indicating end trial
        elapsedTime2(k,1)   = toc;                                         % Store
stimulus time
        reactionTime(k,1)  = rt;                                           % Store
reaction time
        guidata(hObject,handles);                                          % Update
GUI data
    end
    pulseEndExperiment2       = 's40';
    ClockEndExperiment2       = clock;

    fprintf(super.s,pulseEndExperiment2);                                 % End of
trial

    super.reactionTime_Experiment2  = reactionTime;                        % Write
reaction times in super
    super.ClockCross_Experiment2    = ClockCross;                          % Write
control slide times in super
    super.ClockProblem_Experiment2  = ClockProblem;                        % Write
stimulus slide times in super
    super.elapsedTime_Experiment2   = elapsedTime2;                        % Write
trial times in super
```

```
    super.StartExperiment2          = ClockStartExperiment2;
    super.EndExperiment2            = ClockEndExperiment2;

%% -------------------------Start Instruction 3-------------------------
%%

    img_Instruction        =
imread([handles.InstructionMap,char(handles.InstructionFileNames(3))]);
    imshow(img_Instruction);
    getkeywait(9000);
    ClockStartExperiment3(1,:)  = clock;
    pulseStartExperiment3       = 's50';
    fprintf(super.s,pulseStartExperiment3);                               % Pulse
indicating start control trial

%% -------------------------Start Moving Ball----------------------------%%

    ControlSlideExperiment3 = handles.ControlFileNames{3};                 % Create
path for the selected image
    img_control             = imread([handles.ControlMap,ControlSlideExperiment3]);
% Preload the control slide
    Trial                   = 1;
    for j = 1 : 3
        for k = 1 : handles.numberOfVideo
            filenumber         = handles.randomOrderVideo(j,k);                 %
Select the video number from the random array
            fullFileName       = handles.VideoFileNames{filenumber};

            pulseControl      =  strcat('s',num2str(j),num2str(filenumber));
                % Compute pulse number start trial. Pulse number corroborates with
multiplication number
            pulseMovie        =  strcat('s',num2str(j+3),num2str(filenumber));
                   % Create path for the selected video

            CurrentVideo       = squeeze(AllVideoFrames(:,:,:,filenumber));
            fprintf('Videos :%s \n',fullFileName)

             % Show Controlslide
            imshow(img_control);
            axis off;
            ClockControlBall(Trial,:) =
clock;                                     % Store computer time of control slide
            fprintf(super.s,pulseControl);                                      %
Pulse indicating start control trial
            pause(10);
            FrameCount         = length(CurrentVideo(1080,1920,:));
            tic
            for nFrames = 1 : FrameCount-1
                CurrentFrame = double(squeeze(CurrentVideo(:,:,nFrames)))/255;
                if nFrames ==1
                    h = imshow_jdw(CurrentFrame);
                    pause(0.005);
                    ClockMoveBall(Trial,:) = clock;                        % Store
computer time of control slide
                    fprintf(super.s,pulseMovie);
                    while toc < T(nFrames)
                    end
                else
                    set(h,'Cdata',CurrentFrame)
                    pause(0.005)
                    while toc < T(nFrames)
                    end
                end
            end
            elapsedTime3(Trial,:) = toc;
            Trial = Trial + 1;
        end
    end
```

```
    pulseEndExperiment3       = 's60';
    ClockEndExperiment3       = clock;

    fprintf(super.s,pulseEndExperiment3);                                 % End of
trial

    super.elapsedTime_Experiment3   = elapsedTime3;                       % Write
trial times in super
    super.ClockControl_Experiment3  = ClockControlBall;                   % Write
control slide times in super
    super.ClockProblem_Experiment3  = ClockMoveBall;                       % Write
stimulus slide times in super
    super.StartExperiment3          = ClockStartExperiment3;
    super.EndExperiment3            = ClockEndExperiment3;

    EndExperiment        =
imread([handles.InstructionMap,char(handles.InstructionFileNames(5))]);
    imshow(EndExperiment);
    escape = getkeywait(900);
    while escape ~= 27
        escape = getkeywait(900);
    end
    close all


end
% --- Executes when figure1 is resized --- %
function figure1_SizeChangedFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
        global super
        opengl hardware
        set(gcf, 'Color', [0 0 0]);                                         %
Black background colour
        set(gcf,'Units','pixels');                                          %
Set figure size in pixels
        set(gcf,'pos',[-1921 1 1920 1080]);                                 %
Change figure location to second screen
        set(gcf,'MenuBar','none','NumberTitle','off','WindowState',...
            'fullscreen');                                                % Turns
off menubar, numbertitle and sets image to fullscreen
        set(gca,'Visible','off');                                           %
Turns off axis in figure
        set(gca,'LooseInset',get(gca,'TightInset'));
        set(gca,'pos',[0 0 1 1]);
end
```

## C.2 ExperimentBEPMainScript.m

See next page.

```
%% Replicatie - Powerref
%  Lars Kooijman  April 2019
%  This code is run to prompt a GUI which can be used to perform the
%  experimen t. The GUI code
%  can be found in Experiment.m
%  The GUI figure can be found in Eclearxperiment.fig

clear all; %#ok<*CLALL>
close all;
clc; opengl hardware
global super

Pilot = 0;
%% 0. Folder locations
rootFolder = pwd;                                                          % Sets
running folder as root folder
if Pilot == 1
    SaveFolder = strcat(rootFolder,'\Data\Pilot');
else
    SaveFolder = strcat(rootFolder,'\Data\Experiment');
end

%% 1. Serialization
% Preparing serial port to send syncpulse in order to synchronize PlusOptix
% data to the start of the experiment in Matlab.

super.s                     = serial('COM4','BaudRate',115200);            %
Defines the serial port at COM4 and sets the transfer rate
fopen(super.s);                                                            % Opens
the serial port

%% 2. Participant Data Trial 1, 2 & 3
%  Here you define at which distance the screen will be placed during trial
%  1, 2 and 3 including the multiplication set that will be used.
prompt                      = {'Enter Participant Number:'
%                                 ,'Enter distance:','Enter Multiplication Set',...
%                                 'Enter distance:', 'Enter Multiplication Set',...
%                                 'Enter distance:', 'Enter Multiplication Set',...
%                                 'Group Number'
                               };                             % Prompt that
requires input
title                       = 'Participant Number';
dims                        = [1 35];
definput                    = {'Participant00'};
answer                      = inputdlg(prompt,title,dims,definput);
% Stores answer in global variable structure


%% 3. Stimuli location
super.ImageMap              = strcat('./Backgrounds/');
super.ControlMap            = strcat('./ControlSlide/');
super.InstructionMap        = strcat('./Instructions/');
super.MultiplicationMap     = strcat('./Multiplications/');
super.SoundMap              = strcat('./Sounds/');
super.VideoMap              = strcat('./Videos/');
ExperimentBEP;

%% 4. Saving Data

filename                    = strcat(answer{1,1},'.mat');
if exist(SaveFolder)
    cd(SaveFolder)
    save(filename, '-struct', 'super');
else
    mkdir(SaveFolder);
    cd(SaveFolder)
    save(filename, '-struct', 'super');
end
```

## C.3 readRaw.m

See next page.

```
function pilotpart001 = readRaw(filename, startRow, endRow)
%IMPORTFILE Import numeric data from a text file as a matrix.
%   PILOTPART001 = IMPORTFILE(FILENAME) Reads data from text file FILENAME
%   for the default selection.
%
%   PILOTPART001 = IMPORTFILE(FILENAME, STARTROW, ENDROW) Reads data from
%   rows STARTROW through ENDROW of text file FILENAME.
%
% Example:
%   pilotpart001 = importfile('pilot_part_001.csv', 1, 15986);
%
%    See also TEXTSCAN.

% Auto-generated by MATLAB on 2018/12/13 09:19:29

%% Initialize variables.
delimiter = ';';
if nargin<=2
    startRow = 2;
    endRow = inf;
end

%% Read columns of data as text:
% For more information, see the TEXTSCAN documentation.
formatSpec =
'%q%*q%*q%*q%q%*q%*q%q%q%q%*q%*q%*q%*q%*q%q%q%q%*q%*q%*q%q%*q%*q%q%q%q%*q%*q%*q%*q%
*q%q%q%q%*q%*q%*q%*q%*q%q%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to the format.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1, 'Delimiter',
delimiter, 'TextType', 'string', 'HeaderLines', startRow(1)-1, 'ReturnOnError',
false, 'EndOfLine', '\r\n');
for block=2:length(startRow)
    frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-startRow(block)+1,
'Delimiter', delimiter, 'TextType', 'string', 'HeaderLines', startRow(block)-1,
'ReturnOnError', false, 'EndOfLine', '\r\n');
    for col=1:length(dataArray)
        dataArray{col} = [dataArray{col};dataArrayBlock{col}];
    end
end

%% Close the text file.
fclose(fileID);

%% Convert the contents of columns containing numeric text to numbers.
% Replace non-numeric text with NaN.
raw = repmat({''},length(dataArray{1}),length(dataArray)-1);
for col=1:length(dataArray)-1
    raw(1:length(dataArray{col}),col) = mat2cell(dataArray{col},
ones(length(dataArray{col}), 1));
end
numericData = NaN(size(dataArray{1},1),size(dataArray,2));

for col=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
    % Converts text in the input cell array to numbers. Replaced non-numeric
    % text with NaN.
    rawData = dataArray{col};
    for row=1:size(rawData, 1)
        % Create a regular expression to detect and remove non-numeric prefixes and
        % suffixes.
```

```
        regexstr = '(?<prefix>.*?)(?<numbers>([-
]*(\d+[\,]*)+[\.]{0,1}\d*[eEdD]{0,1}[-+]*\d*[i]{0,1})|([-
]*(\d+[\,]*)*[\.]{1,1}\d+[eEdD]{0,1}[-+]*\d*[i]{0,1}))(?<suffix>.*)';
        try
            result = regexp(rawData(row), regexstr, 'names');
            numbers = result.numbers;

            % Detected commas in non-thousand locations.
            invalidThousandsSeparator = false;
            if numbers.contains(',')
                thousandsRegExp = '^\d+?(\,\d{3})*\.{0,1}\d*$';
                if isempty(regexp(numbers, thousandsRegExp, 'once'))
                    numbers = NaN;
                    invalidThousandsSeparator = true;
                end
            end
            % Convert numeric text to numbers.
            if ~invalidThousandsSeparator
                numbers = textscan(char(strrep(numbers, ',', '')), '%f');
                numericData(row, col) = numbers{1};
                raw{row, col} = numbers{1};
            end
        catch
            raw{row, col} = rawData{row};
        end
    end
end


%% Replace non-numeric cells with NaN
R = cellfun(@(x) ~isnumeric(x) && ~islogical(x),raw); % Find non-numeric cells
raw(R) = {NaN}; % Replace non-numeric cells

%% Create output variable
pilotpart001 = cell2mat(raw);
```

## C.4 processRawData.m

See next page.

```
%% Replicatie BEP-groep - Powerref
% Matlab script build by Bastiaan Petermeijer to analyse data
% Edited by Lars Kooijman and Joost de Winter

% Organization CSV datafile
% 1. Timestamp              (milliseconds)
% 2. Eye Tracking Left      (1 = yes, 2 = no)
% 3. Pupil Size X Left      (mm)
% 4. Pupil Size Y Left      (mm)
% 5. Pupil Diameter Left    (mm)
% 6. Gaze X Left            (degrees)
% 7. Gaze Y Left            (degrees)
% 8. Refraction Left        (diopters)
% 9. Eye Tracking Right     (1 = yes, 2 = no)
% 10. Pupil Size X Right    (mm)
% 11. Pupil Size Y Right    (mm)
% 12. Pupil Diameter Right  (mm)
% 13. Gaze X Right          (degrees)
% 14. Gaze Y Right          (degrees)
% 15. Refraction Right      (diopters)
% 16. Sync Pulse Column     (1-100)

clear all; %#ok<*CLALL>
close all;clc;
opengl('save','software')

readdata       = 0;

Pilot          = 0;
Experiment     = 1;

PilotData      = 0;
ExperimentData = 1;

%% Initialization
rootFolder   = pwd;
if Pilot == 1
    dataFolder = strcat(rootFolder,'\Metingen PowerRef\Pilot');
    cd(dataFolder);
    fileNames  = dir('*.csv');
elseif Experiment == 1
    dataFolder = strcat(rootFolder,'\Metingen PowerRef\Experiment');
    MatlabData = strcat(rootFolder,'\Metingen Matlab\Experiment');
    cd(dataFolder);
    fileNames  = dir('*.csv');
end
%Contents     = dir(rootFolder);
%fileNames    = dir('*.csv');

if readdata==1
    % Preallocate data
    data         = NaN(((15*60)/0.02)*1.25, 16, length(fileNames));
    fileSize     = zeros(length(fileNames),1);
    %% Read RawData files
    for pp = 2:length(fileNames) %  Skip Participant 1 for sounds
        fprintf('Reading participant %.0d \n',pp);
        temp                          = readRaw(fileNames(pp).name);
        start                         = find(temp(:,16)==10,1);         %
Removes data before experiment starts
        ending                        = find(temp(:,16)==60,1);         %
Renoves data after experiment ended
        fileSize(pp,1)                = length(temp(start:ending,:));
        data(1:fileSize(pp,1),:,pp)      = temp(start:ending,:);
    end
    %% Store imported Data to mat-file
    CutSize = max(fileSize);
    data    = data(1:CutSize,:,:);
    fprintf('----------------- Saving data \n');
```

```
    save('RawData','data','-v7.3');
    cd(rootFolder);

else
    load RawData
    cd(rootFolder);
    fileNames  = dir('*.mat');

end
%% Blink and missing data removal
% Variabes 1, 2, 9 and 16 do not need to be interpolated because they are
% binary.
datafix         = data(:,:,2:36);% participant 1 is NANS
datafilt        = data(:,:,2:36);
fc              = 4;
fs              = 50;
[b,a]           = butter(3,(fc/(fs/2)),'low');
for pp = 1 : 35                                                            % Loop
over all files
    for nn = [3 4 5 6 7 8 10 11 12 13 14 15]                               % For
these files a 0 is missing data
        temp    = squeeze(data(:,nn,pp+1));
        temp(isnan(temp)) = [];
        temp(temp == 0 | temp == -100) = NaN;
        temp1   = squeeze(data(:,6,pp+1));                                   %
6.  Gaze X Left           (degrees)
        temp2   = squeeze(data(:,7,pp+1));                                   %
7.  Gaze Y Left           (degrees)
        temp3   = squeeze(data(:,13,pp+1));                                  % 13.
Gaze X Right          (degrees)
        temp4   = squeeze(data(:,14,pp+1));                                  % 14.
Gaze Y Right          (degrees)

        temp(temp1<-30 | temp2<-30 | temp3<-30 | temp4<-30 |...
             temp1>30 | temp2>30 | temp3>30 | temp4>30) = NaN;             % Remove
data if there are eye movements
        MM(pp,nn)    = mean(isnan(temp));
        blinkmoments = FindBlink(temp,2);                                  % Find
the blinks with period before and after each blink (remove 5 samples, 0.1 s before
and after)
        temp(blinkmoments)=NaN;
        if sum(~isnan(temp))>0
            temp=fixgapsj(temp); % interpolate the data around blinks
            datafix(1:length(temp),nn,pp)=temp;
            datafilt(1:length(temp),nn,pp)=filtfilt(b,a,temp);
        else
            datafix(:,nn,pp)=NaN;
            datafilt(:,nn,pp)=NaN;                                            %
Tijd,Variabele,Deelnemers
        end
    end
end

%% sort participants on Task 1: Vergence and accommodation check
MatrixTask1_1         = NaN(((10)/0.02)*1.05, 6, length(datafilt(1,1,:))); % 1 toon
duurt 10s
MatrixTask1_2         = NaN(((10)/0.02)*1.05, 6, length(datafilt(1,1,:))); % /0.02
betekent 50 samples per seconde
                                                                           % 1.05
is 5% marge van de dataset
                                                                           % 6
tonen (3 laag, 3 hoog)
StartPulseTask1 = 10;
EndPulseTask1   = 20;

% AANPASSEN WELKE KOLOMMEN WE WILLEN VOOR DE ANALYSE!
DataTask1_1       = cell(2,12);
DataTask1_2       = cell(2,12);
```

```
DataTask1_1{1,1}   = 'Pupil Size X Left';     %3
DataTask1_1{1,2}   = 'Pupil Size Y Left';     %4
DataTask1_1{1,3}   = 'Pupil Diameter Left';   %5    <--
DataTask1_1{1,4}   = 'Gaze X Left';           %6   (<--)
DataTask1_1{1,5}   = 'Gaze Y Left';           %7    <--
DataTask1_1{1,6}   = 'Refraction Left';       %8    <--

DataTask1_1{1,7}   = 'Pupil Size X Right';    %10
DataTask1_1{1,8}   = 'Pupil Size Y Right';    %11
DataTask1_1{1,9}   = 'Pupil Diameter Right';  %12   <--
DataTask1_1{1,10}  = 'Gaze X Right';          %13  (<--)
DataTask1_1{1,11}  = 'Gaze Y Right';          %14   <--
DataTask1_1{1,12}  = 'Refraction Right';      %15   <--

DataTask1_2{1,1}   = 'Pupil Size X Left';     %3
DataTask1_2{1,2}   = 'Pupil Size Y Left';     %4
DataTask1_2{1,3}   = 'Pupil Diameter Left';   %5
DataTask1_2{1,4}   = 'Gaze X Left';           %6
DataTask1_2{1,5}   = 'Gaze Y Left';           %7
DataTask1_2{1,6}   = 'Refraction Left';       %8

DataTask1_2{1,7}   = 'Pupil Size X Right';    %10
DataTask1_2{1,8}   = 'Pupil Size Y Right';    %11
DataTask1_2{1,9}   = 'Pupil Diameter Right';  %12
DataTask1_2{1,10}  = 'Gaze X Right';          %13
DataTask1_2{1,11}  = 'Gaze Y Right';          %14
DataTask1_2{1,12}  = 'Refraction Right';      %15

for nParticipant = 1 : 35
    StartLoopHigh  = StartPulseTask1;
    EndLoopHigh    = EndPulseTask1;
    StartPointLoop = find(datafilt(:,16,nParticipant) == StartLoopHigh,1);
    EndPointLoop   = find(datafilt(:,16,nParticipant) == EndLoopHigh,1);
    templengthlow  = StartPointLow+1:EndPointLow;
    MatrixTask1_1(1:length(templengthlow),nTone,nParticipant) =
datafilt(StartPointLow+1:EndPointLow,nVariables,nParticipant);

end


k = 1;
for nVariables = [3 4 5 6 7 8 10 11 12 13 14 15]
    for nParticipant = 1 : 35
        for nTone = 1 : 3
            if nParticipant == 17 && nTone == 1 && nVariables == 3
                templength = length(datafilt(3717:end,16,nParticipant)); %
Experiment restarted, cut off first bit.
                datafilt(1:templength,:,nParticipant) =
datafilt(3717:end,:,nParticipant);
            end
            % Low tone pulses
            StartToneLow     = nTone;
            EndToneLow       = nTone + 90;
            % High tone pulses
            StartToneHigh    = nTone + 90;
            EndToneHigh      = nTone+1;

%             TaskLimit1       = find(datafilt(:,16,nParticipant) ==
StartPulseTask1,1);

            StartPointLow    = find(datafilt(:,16,nParticipant) == StartToneLow,1);
            StartPointHigh   = find(datafilt(:,16,nParticipant) ==
StartToneHigh,1);
            EndPointLow      = find(datafilt(:,16,nParticipant) == EndToneLow,1);
%             EndPointHigh     = find(datafilt(:,16,nParticipant) ==
EndToneHigh,1);
            if nTone < 3
```

```
                EndPointHigh = find(datafilt(:,16,nParticipant) == EndToneHigh,1);
            else
                EndPointLow = find(datafilt(:,16,nParticipant) == 93,1);
                EndPointHigh = find(datafilt(:,16,nParticipant) == 20,1);
            end
            templengthlow  = StartPointLow+1:EndPointLow;
            templengthhigh = StartPointHigh+1:EndPointHigh;
            MatrixTask1_1(1:length(templengthlow),nTone,nParticipant) =
datafilt(StartPointLow+1:EndPointLow,nVariables,nParticipant);   %Check of
eventueel nog +1
            MatrixTask1_2(1:length(templengthhigh),nTone,nParticipant) =
datafilt(StartPointHigh+1:EndPointHigh,nVariables,nParticipant); %Check of
eventueel nog +1
        end
    end
DataTask1_1{2,k} = MatrixTask1_1;
DataTask1_2{2,k} = MatrixTask1_2;
k = k + 1;
end

%% sort participants on Task 2: Multiplications
MatrixTask2         = NaN(ceil(((25)/0.02)*1.05), 9, length(datafilt(1,1,:)));   %
1 som duurt 25s // ceil is nodig omdat de smaples anders geen heel getal zijn, maar
1312,5. nu is het 1313
                                                                              % /0.02
betekent 50 samples per seconde
                                                                              % 1.05
is 5% marge van de dataset
                                                                              % 9
multiplications
StartPulseTask2 = 30;
EndPulseTask2   = 40;

% AANPASSEN WELKE KOLOMMEN WE WILLEN VOOR DE ANALYSE!
DataTask2       = cell(2,12);
DataReaction    = cell(9,35);

DataTask2{1,1}   = 'Pupil Size X Left';      %3
DataTask2{1,2}   = 'Pupil Size Y Left';      %4
DataTask2{1,3}   = 'Pupil Diameter Left';    %5
DataTask2{1,4}   = 'Gaze X Left';            %6
DataTask2{1,5}   = 'Gaze Y Left';            %7
DataTask2{1,6}   = 'Refraction Left';        %8

DataTask2{1,7}   = 'Pupil Size X Right';     %10
DataTask2{1,8}   = 'Pupil Size Y Right';     %11
DataTask2{1,9}   = 'Pupil Diameter Right';   %12
DataTask2{1,10}  = 'Gaze X Right';           %13
DataTask2{1,11}  = 'Gaze Y Right';           %14
DataTask2{1,12}  = 'Refraction Right';       %15

k = 1;
cd(MatlabData)
for nVariables = [3 4 5 6 7 8 10 11 12 13 14 15]
    for nParticipant = 1 : 35
        if nParticipant <9 && nVariables == 3
            load(strcat('Participant0',num2str(nParticipant+1),'.mat'))
        elseif nParticipant >= 9 && nVariables == 3
            load(strcat('Participant',num2str(nParticipant+1),'.mat'))
        end
        for nMultiplication = 1 : 9
            StartSum         = nMultiplication + 70;
            ReactionSum      = nMultiplication + 80;
            EndSum           = nMultiplication + 90;
            TaskLimit2       = find(datafilt(:,16,nParticipant) ==
StartPulseTask2,1);
```

```
            StartPointSum    = find(datafilt(TaskLimit2:end,16,nParticipant) ==
StartSum,1);
            ReactionPointSum = find(datafilt(TaskLimit2:end,16,nParticipant) ==
ReactionSum,1);
            EndPointSum      = find(datafilt(TaskLimit2:end,16,nParticipant) ==
EndSum,1);

            datalengthtask2  = TaskLimit2+StartPointSum+1:TaskLimit2+EndPointSum;
            MatrixTask2(1:length(datalengthtask2),nMultiplication,nParticipant) =
datafilt(TaskLimit2+StartPointSum+1:TaskLimit2+EndPointSum,nVariables,nParticipant)
; %Check of eventueel nog +1
            LocationRT       = RandomOrderMultiplications==nMultiplication;
            ReactionTime(nMultiplication,nParticipant) =
reactionTime_Experiment2(LocationRT);
%             DataReaction(nMultiplication,
        end
    end
DataTask2{2,k} = MatrixTask2;
% DataReaction(
k = k + 1;
end
cd(rootFolder);

%% sort participants on Task 3: Follow the ball
MatrixTask3         = NaN(((16)/0.02)*1.05, 24, length(datafilt(1,1,:)));   % 1
video duurt 16s
                                                                            % /0.02
betekent 50 samples per seconde
                                                                            % 1.05
is 5% marge van de dataset
                                                                            % 24
videos (3 loops van 8 videos)
StartPulseTask3 = 50;
EndPulseTask3   = 60;

% AANPASSEN WELKE KOLOMMEN WE WILLEN VOOR DE ANALYSE!
DataTask3       = cell(2,12);

DataTask3{1,1}   = 'Pupil Size X Left';     %3
DataTask3{1,2}   = 'Pupil Size Y Left';     %4
DataTask3{1,3}   = 'Pupil Diameter Left';   %5
DataTask3{1,4}   = 'Gaze X Left';           %6
DataTask3{1,5}   = 'Gaze Y Left';           %7
DataTask3{1,6}   = 'Refraction Left';       %8

DataTask3{1,7}   = 'Pupil Size X Right';    %10
DataTask3{1,8}   = 'Pupil Size Y Right';    %11
DataTask3{1,9}   = 'Pupil Diameter Right';  %12
DataTask3{1,10}  = 'Gaze X Right';          %13
DataTask3{1,11}  = 'Gaze Y Right';          %14
DataTask3{1,12}  = 'Refraction Right';      %15

k = 1;
for nVariables = [3 4 5 6 7 8 10 11 12 13 14 15]
    for nParticipant = 1 : 35
        for j = 1 : 3
            for nVideo = 1 : 8
                StartVideo         = nVideo + 10*j;
                EndVideo           = nVideo + 10*j + 30;
                TaskLimit3         = find(datafilt(:,16,nParticipant) ==
StartPulseTask3,1);

                StartPointVideo    = find(datafilt(TaskLimit3:end,16,nParticipant)
== StartVideo,1);
                VideoPulses        = find(datafilt(TaskLimit3:end,16,nParticipant)
== EndVideo);
                EndPointVideo      = VideoPulses(1,1)+length(VideoPulses);
```

```
                datalengthtask3  =
TaskLimit3+StartPointVideo+1:TaskLimit3+EndPointVideo;
                MatrixTask3(1:length(datalengthtask3),nVideo+8*j-8,nParticipant) =
datafilt(TaskLimit3+StartPointVideo+1:TaskLimit3+EndPointVideo,nVariables,nParticip
ant); %Check of eventueel nog +1
            end
        end
    end
DataTask3{2,k} = MatrixTask3;
k = k + 1;
end

%% Save FiltData in folders
if PilotData == 1
    SaveFolder = strcat(rootFolder,'\FiltData\Pilot');
elseif ExperimentData == 1
    SaveFolder = strcat(rootFolder,'\FiltData\Experiment');
end

%% Save Organized Data
filename1_1           = strcat('FiltDataSoundsLow.mat');
filename1_2           = strcat('FiltDataSoundsHigh.mat');
filename2             = strcat('FiltDataMultiplications.mat');
filename3             = strcat('FiltDataVideos.mat');
if exist(SaveFolder)
    cd(SaveFolder)
    save(filename1_1, 'DataTask1_1', '-v7.3');
    save(filename1_2, 'DataTask1_2', '-v7.3');
    save(filename2, 'DataTask2', '-v7.3');
    save(filename3, 'DataTask3', '-v7.3');
else
    mkdir(SaveFolder);
    cd(SaveFolder)
    save(filename1_1, 'DataTask1_1', '-v7.3');
    save(filename1_2, 'DataTask1_2', '-v7.3');
    save(filename2, 'DataTask2', '-v7.3');
    save(filename3, 'DataTask3', '-v7.3');
end

%% mean van soort en standaard deviatie

%% plot pupilsize
```

## C.5 StaticsFiltDatapaper.m

See next page.

```
%% Replicatie BEP-groep - Powerref
% Matlab script build by Lars Kooijman to analyse data
% Edited by Stefan Jansen, Julia Russell and Tamir Themans
% Scatter plots & Cross correlations(from line 1160)


clear all; %#ok<*CLALL>
close all;clc;
opengl('save','software')

PilotData       = 0;
ExperimentData  = 1;

%% Initialization
rootFolder   = pwd;
if PilotData == 1
    dataFolder = strcat(rootFolder,'\FiltData\Pilot');
    cd(dataFolder);
    load FiltDataSoundsLow
    load FiltDataSoundsHigh
    load FiltDataMultiplications
    load FiltDataVideos
%     fileNames  = dir('*.mat');

elseif ExperimentData == 1
    dataFolder = strcat(rootFolder,'\FiltData\Experiment');
    cd(dataFolder);
    load FiltDataSoundsLow
    load FiltDataSoundsHigh
    load FiltDataMultiplications
    load FiltDataVideos


end

%% Variables

% 1.  Pupil Size X Left
% 2.  Pupil Size Y Left
% 3.  Pupil Diameter Left    <--    PDL
% 4.  Gaze X Left            <--    VL (X)
% 5.  Gaze Y Left            <--    VL (Y)
% 6.  Refraction Left        <--    AL
%
% 7.  Pupil Size X Right
% 8.  Pupil Size Y Right
% 9.  Pupil Diameter Right   <--    PDR
% 10. Gaze X Right           <--    VR (X)
% 11. Gaze Y Right           <--    VR (Y)
% 12. Refraction Right       <--    AR


% Task 1: PDL, VL (X), AL
%         PDR, VR (X), AR
% Task 2: PDL, AL
%         PDR, AR
% Task 3: PDL, VL (Y), AL
%         PDR, VR (Y), AR

%% Colors for plots
colors = [204 255 153;        % 1 lichtgroen
          153 255 51 ;        % 2 middelgroen
          102 204 0  ;        % 3 donkergroen
          153 153 255;        % 4 lichtblauw
          51  51  255;        % 5 middelblauw
          0   0   204;        % 6 donkerblauw
          255 153 153;        % 7 lichtrood
          255 51  51 ;        % 8 middelrood
```

```
          204 0   0  ;        % 9 donkerrood
          255 102 178;        % 10 lichtroze
          255 0   127;        % 11 middelroze
          153 0   76 ;        % 12 donkerroze
          153 0   76 ;        % 13 donkerpaars
          0   255 255;        % 14 turquiose
          0   128 255;        % 15 donkerturquiose
          0   0   0  ]/255;   % 16 zwart

%% DATA CHECKEN OP NaN's PER ONDERDEEL

% Task 1
VarianceMatrix = NaN(6,35,8);
nLow    = 1;
nHigh   = 3;
Variables  = [3 4 5 6 9 10 11 12];

for nVariables = [3 4 5 6 9 10 11 12]
    for nParticipant = 1 : 35
        nLow    = 1;
        nHigh   = 4;
        for nTonesLow = 1 : 3
            TempLowTones  = DataTask1_1{2,nVariables}(:,nTonesLow,nParticipant);
            VarianceLow   = nanvar(TempLowTones);
            VarianceMatrix(nLow,nParticipant,(nVariables == Variables)) =
VarianceLow;
            nLow = nLow + 1;
        end
        for nTonesHigh = 1 : 3
            TempHighTones = DataTask1_2{2,nVariables}(:,nTonesHigh,nParticipant);
            VarianceHigh  = nanvar(TempHighTones);
            VarianceMatrix(nHigh,nParticipant,(nVariables == Variables)) =
VarianceHigh;
            nHigh = nHigh + 1;
        end
    end
end


%% -------------------------  TASK 3  ---------------------------------- %%

% PDL3 = Pupil Diameter Right, Task 3
% VL3  = Gaze Y Right, Task 3
% AL3  = Refraction Right, Task 3
% PDR3 = Pupil Diameter Right, Task 3
% VR3  = Gaze Y Right, Task 3
% AR3  = Refraction Right, Task 3

% Inladen variabelen
DataPDL3 = DataTask3{2,3};
DataVL3  = DataTask3{2,5};
DataAL3  = DataTask3{2,6};

DataPDR3 = DataTask3{2,9};
DataVR3  = DataTask3{2,11};
DataAR3  = DataTask3{2,12};

k = 1;
for nVideos = 1:24
    PDL3          = squeeze(DataPDL3(:,nVideos,:));
    VL3           = squeeze(DataVL3(:,nVideos,:));
    AL3           = squeeze(DataAL3(:,nVideos,:));

    PDR3          = squeeze(DataPDR3(:,nVideos,:));
    VR3           = squeeze(DataVR3(:,nVideos,:));
    AR3           = squeeze(DataAR3(:,nVideos,:));

    PDL3_Row{1,k} = PDL3;
```

```
    VL3_Row{1,k}  = VL3;
    AL3_Row{1,k}  = AL3;
    PDR3_Row{1,k} = PDR3;
    VR3_Row{1,k}  = VR3;
    AR3_Row{1,k}  = AR3;
    k = k + 1;
end
%% (SUBQUESTION 2+3+4)
% Subquestion (2+3+4): comparing the 4 levels with eachother
Indices = [1:4:21;
           2:4:22;
           3:4:23;
           4:4:24];

for n = 1:4                                                            % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    PDL3_Temp            = DataPDL3(:,Indices(n,:),:);
    PDR3_Temp            = DataPDR3(:,Indices(n,:),:);
    PDL3_MeanMatrix{n,1} = squeeze(nanmean(PDL3_Temp,2));
    PDR3_MeanMatrix{n,1} = squeeze(nanmean(PDR3_Temp,2));
    PD3_MeanMatrixGem    = (PDL3_MeanMatrix{n,1} + PDR3_MeanMatrix{n,1})/2;
    PD3_MeanMatrix{n,1}  = PD3_MeanMatrixGem;
end

for n = 1:4                                                            % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    VL3_Temp            = DataVL3(:,Indices(n,:),:);
    VR3_Temp            = DataVR3(:,Indices(n,:),:);
    VL3_MeanMatrix{n,1} = squeeze(nanmean(VL3_Temp,2));
    VR3_MeanMatrix{n,1} = squeeze(nanmean(VR3_Temp,2));
    V3_MeanMatrixGem    = (VL3_MeanMatrix{n,1} + VR3_MeanMatrix{n,1})/2;
    V3_MeanMatrix{n,1}  = V3_MeanMatrixGem;
end

for n = 1:4                                                            % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    AL3_Temp            = DataAL3(:,Indices(n,:),:);
    AR3_Temp            = DataAR3(:,Indices(n,:),:);
    AL3_MeanMatrix{n,1} = squeeze(nanmean(AL3_Temp,2));
    AR3_MeanMatrix{n,1} = squeeze(nanmean(AR3_Temp,2));
    A3_MeanMatrixGem    = (AL3_MeanMatrix{n,1} + AR3_MeanMatrix{n,1})/2;
    A3_MeanMatrix{n,1}  = A3_MeanMatrixGem;
end


% PD3C = Pupil Diameter Change in Percentage
% V3C = Vergence, Gaze Y Change in Percentage
% A3C = Accommodation, refraction Change in Percentage
PD3C_Level = NaN(35,4);
for nLevel = 1:4
    for nParticipants = 1 : 35
        MeanControlSlidePD234 =
nanmean(PD3_MeanMatrix{nLevel,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoPD234         =
max(PD3_MeanMatrix{nLevel,1}(50*10.5:50*15.5,nParticipants));
        PD3C_LevelX       = (MaxVideoPD234 -
MeanControlSlidePD234)/MeanControlSlidePD234;
        PD3C_Level(nParticipants,nLevel) = PD3C_LevelX*100;            % kolom
is level 1, kolom 2 is level 2, enz.
    end
end

V3C_Level = NaN(35,4);
for nLevel = 1:4
    for nParticipants = 1 : 35
        MeanControlSlideV234 =
nanmean(V3_MeanMatrix{nLevel,1}(50*0.5:50*9.5,nParticipants));
```

```
        MaxVideoV234         =
max(V3_MeanMatrix{nLevel,1}(50*10.5:50*15.5,nParticipants));
        V3C_LevelX       = (MaxVideoV234 -
MeanControlSlideV234)/MeanControlSlideV234;
        V3C_Level(nParticipants,nLevel) = V3C_LevelX*100;                    % kolom
is level 1, kolom 2 is level 2, enz.
    end
end

A3C_Level = NaN(35,4);
for nLevel = 1:4
    for nParticipants = 1 : 35
        MeanControlSlideA234 =
nanmean(A3_MeanMatrix{nLevel,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoA234         =
max(A3_MeanMatrix{nLevel,1}(50*10.5:50*15.5,nParticipants));
        A3C_LevelX       = (MaxVideoA234 -
MeanControlSlideA234)/MeanControlSlideA234;
        A3C_Level(nParticipants,nLevel) = A3C_LevelX*100; % kolom is level 1, kolom
2 is level 2, enz.
    end
end

% PD3C_Level (35 participanten x 4 levels)
% V3C_Level (35 participanten x 4 levels)
% A3C_Level (35 participanten x 4 levels)
%% LEVEL 1 PD3 <-> V3 (SUBQUESTION 2+3+4)
% PD3 <-> V3
%

fitA = PD3C_Level(:,1);     % X-as
fitB = V3C_Level(:,1);      % Y-as
for n=1:35
    if  (fitB(n,1)< -500) || (fitB(n,1)> 500)
            fitB(n,1)  = NaN;
            fitA(n,1)  =NaN;
    end
end
for n=1:35
    if  fitA(n,1)<-1 || fitA(n,1)>7
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 1');
xlabel('Pupil diameter change [%]');
ylabel('Gaze(Y) change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
V_Level1.jpg')                         %uncomment to save figure
%% LEVEL 2 PD3 <-> V3 (SUBQUESTION 2+3+4)
% PD3 <-> V3
fitA = PD3C_Level(:,2);     % X-as
```

```
fitB = V3C_Level(:,2);      % Y-as
for n=1:35
    if  (fitB(n,1)< -500) || (fitB(n,1)> 500)
            fitB(n,1)  = NaN;
            fitA(n,1)  =NaN;
    end
end
for n=1:35
    if  fitA(n,1)<0.2754 || fitA(n,1)>4.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 2');
xlabel('Pupil diameter change [%]');
ylabel('Gaze(Y) change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
V_Level2.jpg')                        %uncomment to save figure
%% LEVEL 3 PD3 <-> V3 (SUBQUESTION 2+3+4)
% PD3 <-> V3
fitA = PD3C_Level(:,3);     % X-as
fitB = V3C_Level(:,3);      % Y-as
for n=1:35
    if  fitA(n,1)<0.3786 || fitA(n,1)>5.0204
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-494.7744 || fitB(n,1)>177.3608
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 3');
xlabel('Pupil Diameter Change [%]');
ylabel('Gaze(Y) Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
```

```
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-V_Level3.jpg')
%% LEVEL 4 PD3 <-> V3 (SUBQUESTION 2+3+4)
% PD3 <-> V3
fitA = PD3C_Level(:,4);     % X-as
fitB = V3C_Level(:,4);      % Y-as
for n=1:35
    if  fitA(n,1)<0.2070 || fitA(n,1)>4.1950
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-433.1405 || fitB(n,1)>145.9765
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 4');
xlabel('Pupil Diameter Change [%]');
ylabel('Gaze(Y) Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);

%saveas(Scatterplot,'Scatterplot_Task_3_PD-V_Level4.jpg')

%% LEVEL 1 PD3 <-> A3 (SUBQUESTION 2+3+4)
% PD3 <-> A3
fitA = PD3C_Level(:,1);     % X-as
fitB = A3C_Level(:,1);      % Y-as

for n=1:35
    if  fitB(n,1)<-50 ||  fitB(n,1)>30
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 1')
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
```

```
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-A_Level1.jpg')

%% LEVEL 2 PD3 <-> A3 (SUBQUESTION 2+3+4)
% PD3 <-> A3
fitA = PD3C_Level(:,2);     % X-as
fitB = A3C_Level(:,2);      % Y-as
for n=1:35
    if  fitA(n,1)<0.2754 || fitA(n,1)>4.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-100 || fitB(n,1)>100
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 2')
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-A_Level2.jpg')
%% LEVEL 3 PD3 <-> A3 (SUBQUESTION 2+3+4)
% PD3 <-> A3
fitA = PD3C_Level(:,3);     % X-as
fitB = A3C_Level(:,3);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>4.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-94.5603|| fitB(n,1)>49.5153
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
```

```
title('Scatterplot of Task 3: level 3')
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-A_Level3.jpg')
%% LEVEL 4 PD3 <-> A3 (SUBQUESTION 2+3+4)
% PD3 <-> A3
fitA = PD3C_Level(:,4);     % X-as
fitB = A3C_Level(:,4);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>4.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-81.5603|| fitB(n,1)>49.5153
            fitB(n,1)  = NaN;
            fitA(n,1)  = NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 4')
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-A_Level4.jpg')

%% LEVEL 1 V3 <-> A3 (SUBQUESTION 2+3+4)
% V3 <-> A3
fitA = V3C_Level(:,1);     % X-as
fitB = A3C_Level(:,1);     % Y-as
for n=1:35
    if fitA(n,1)<-500 || fitA(n,1)>500
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-50|| fitB(n,1)>50
            fitB(n,1) =  NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
```

```
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'g-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 1')
xlabel('Gaze (Y) Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_V-A_Level1.jpg')

%% LEVEL 2 V3 <-> A3 (SUBQUESTION 2+3+4)
% V3 <-> A3
fitA = V3C_Level(:,2);     % X-as
fitB = A3C_Level(:,2);     % Y-as
for n=1:35
    if fitA(n,1)<-475.8273 || fitA(n,1)>127.7355
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-60.8422 || fitB(n,1)>67.66
            fitB(n,1) =  NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'g-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 2')
xlabel('Gaze(Y) Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_V-A_Level2.jpg')

%% LEVEL 3 V3 <-> A3 (SUBQUESTION 2+3+4)
% V3 <-> A3
fitA = V3C_Level(:,3);     % X-as
fitB = A3C_Level(:,3);      % Y-as
for n=1:35
    if fitA(n,1)<-475.8273 || fitA(n,1)>127.7355
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-60.8422 || fitB(n,1)>67.66
            fitB(n,1) =  NaN;
            fitA(n,1) =  NaN
    end
end
```

```
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'g-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 3')
xlabel('Gaze(Y) Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_V-A_Level3.jpg')

%% LEVEL 4 V3 <-> A3 (SUBQUESTION 2+3+4)
% V3 <-> A3
fitA = V3C_Level(:,4);     % X-as
fitB = A3C_Level(:,4);     % Y-as
for n=1:35
    if fitA(n,1)<-475.8273 || fitA(n,1)>127.7355
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-60.8422 || fitB(n,1)>67.66
            fitB(n,1) =  NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'g-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: level 4');
xlabel('Gaze(Y) Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_V-A_Level4.jpg')

%% SUBQUETION 1
% Subquestion (1): comparing dynamic vs static.
Indices1 = [1:4 9:12 17:20;     %dynamisch
           5:8 13:16 21:24];    %statisch


for n = 1:2                  % links en recht gemiddeld, van alle statische video's,
en alle dynamische video's. in totaal 2 groepen.
    PDL3_Temp1            = DataPDL3(:,Indices1(n,:),:);
    PDR3_Temp1            = DataPDR3(:,Indices1(n,:),:);
```

```
    PDL3_MeanMatrix1{n,1} = squeeze(nanmean(PDL3_Temp1,2));
    PDR3_MeanMatrix1{n,1} = squeeze(nanmean(PDR3_Temp1,2));
    PD3_MeanMatrixGem    = (PDL3_MeanMatrix1{n,1} + PDR3_MeanMatrix1{n,1})/2;
    PD3_MeanMatrix1{n,1}  = PD3_MeanMatrixGem;
end

for n = 1:2                 % links en recht gemiddeld, van alle statische video's,
en alle dynamische video's. in totaal 2 groepen.
    VL3_Temp1            = DataVL3(:,Indices1(n,:),:);
    VR3_Temp1            = DataVR3(:,Indices1(n,:),:);
    VL3_MeanMatrix1{n,1} = squeeze(nanmean(VL3_Temp1,2));
    VR3_MeanMatrix1{n,1} = squeeze(nanmean(VR3_Temp1,2));
    V3_MeanMatrixGem1    = (VL3_MeanMatrix1{n,1} + VR3_MeanMatrix1{n,1})/2;
    V3_MeanMatrix1{n,1}  = V3_MeanMatrixGem1;
end

for n = 1:2                 % links en recht gemiddeld, van alle statische video's,
en alle dynamische video's. in totaal 2 groepen.
    AL3_Temp1            = DataAL3(:,Indices1(n,:),:);
    AR3_Temp1            = DataAR3(:,Indices1(n,:),:);
    AL3_MeanMatrix1{n,1} = squeeze(nanmean(AL3_Temp1,2));
    AR3_MeanMatrix1{n,1} = squeeze(nanmean(AR3_Temp1,2));
    A3_MeanMatrixGem1    = (AL3_MeanMatrix1{n,1} + AR3_MeanMatrix1{n,1})/2;
    A3_MeanMatrix1{n,1}  = A3_MeanMatrixGem1;
end


% PD3C = Pupil Diameter Change in Percentage
% V3C = Vergence, Gaze Y Change in Percentage
% A3C = Accommodation, refraction Change in Percentage
PD3C_Level1 = NaN(35,2);
for nLevel1 = 1:2
    for nParticipants = 1 : 35
        MeanControlSlidePD1 =
nanmean(PD3_MeanMatrix1{nLevel1,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoPD1         =
max(PD3_MeanMatrix1{nLevel1,1}(50*10.5:50*15.5,nParticipants));
        PD3C_Level1X       = (MaxVideoPD1 -
MeanControlSlidePD1)/MeanControlSlidePD1;
        PD3C_Level1(nParticipants,nLevel1) = PD3C_Level1X*100;                 %
kolom 1 is dynamisch, kolom 2 is statisch?
    end
end

V3C_Level1 = NaN(35,2);
for nLevel1 = 1:2
    for nParticipants = 1 : 35
        MeanControlSlideV1 =
nanmean(V3_MeanMatrix1{nLevel1,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoV1         =
max(V3_MeanMatrix1{nLevel1,1}(50*10.5:50*15.5,nParticipants));
        V3C_Level1X       = (MaxVideoV1 - MeanControlSlideV1)/MeanControlSlideV1;
        V3C_Level1(nParticipants,nLevel1) = V3C_Level1X*100;                   %
kolom 1 is dynamisch, kolom 2 is statisch?
    end
end

A3C_Level1 = NaN(35,2);
for nLevel1 = 1:2
    for nParticipants = 1 : 35
        MeanControlSlideA1 =
nanmean(A3_MeanMatrix1{nLevel1,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoA1         =
max(A3_MeanMatrix1{nLevel1,1}(50*10.5:50*15.5,nParticipants));
        A3C_Level1X       = (MaxVideoA1 - MeanControlSlideA1)/MeanControlSlideA1;
        A3C_Level1(nParticipants,nLevel1) = A3C_Level1X*100;               % kolom
1 is dynamisch, kolom 2 is statisch?
    end
```

```
end

%% DYNAMIC PD3 <-> V3 (SUBQUESTION 1)
% PD3 <-> V3
fitA = PD3C_Level1(:,1);     % X-as
fitB = V3C_Level1(:,1);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>5.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-300.8422 || fitB(n,1)>100
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: Dynamic');
xlabel('Pupil Diameter Change [%]');
ylabel('Gaze(Y) Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
V_Dynamic.jpg')                         %uncomment to save figure

%% STATIC PD3 <-> V3 (SUBQUESTION 1)
% PD3 <-> V3
fitA = PD3C_Level1(:,2);     % X-as
fitB = V3C_Level1(:,2);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>5.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-300.8422 || fitB(n,1)>100
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: Static');
```

```
xlabel('Pupil Diameter Change [%]');
ylabel('Gaze(Y) Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
V_Static.jpg')                        %uncomment to save figure

%% DYNAMIC PD3 <-> A3 (SUBQUESTION 1)
% PD3 <-> A3
fitA = PD3C_Level1(:,1);     % X-as
fitB = A3C_Level1(:,1);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>5.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-80 || fitB(n,1)>50
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3:Dynamic');
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
A_Dynamic.jpg')                        %uncomment to save figure

%% Static PD3 <-> A3 (SUBQUESTION 1)
% PD3 <-> A3
fitA = PD3C_Level1(:,2);     % X-as
fitB = A3C_Level1(:,2);      % Y-as
for n=1:35
    if fitA(n,1)<0.2754 || fitA(n,1)>5.5688
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-80 || fitB(n,1)>50
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
```

```
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: Static');
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_PD-
A_Static.jpg')                        %uncomment to save figure

%% DYNAMIC V3 <-> A3 (SUBQUESTION 1)
% V3 <-> A3
fitA = V3C_Level1(:,1);      % X-as
fitB = A3C_Level1(:,1);      % Y-as
for n=1:35
    if fitA(n,1)<-400 || fitA(n,1)>0
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB(n,1)<-50 || fitB(n,1)>50
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'r-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: Dynamic');
xlabel('Gaze(Y) change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
saveas(Scatterplot,'Scatterplot_Task_3_V-
A_Dynamic.jpg')                        %uncomment to save figure


%% STATIC V3 <-> A3 (SUBQUESTION 1)
% V3 <-> A3
fitA = V3C_Level1(:,2);      % X-as
fitB = A3C_Level1(:,2);      % Y-as
for n=1:35
    if fitA(n,1)<-400 || fitA(n,1)>0
            fitA(n,1)  = NaN;
            fitB(n,1)  = NaN;
    end
end
for n=1:35
```

```
    if  fitB(n,1)<-50 || fitB(n,1)>50
            fitB(n,1)  = NaN;
            fitA(n,1) =  NaN
    end
end
isnan(fitA(:,1))
VectorNaN = isnan(fitA(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA(isnan(fitA))=[];
fitB(isnan(fitB))=[];
scatter(fitA,fitB,35,'b','*')
P = polyfit(fitA,fitB,1);
FitLine = P(1)*fitA+P(2); hold on;
Scatterplot = plot(fitA,FitLine,'m-','LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 3: Static');
xlabel('Gaze(Y) Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1);
%saveas(Scatterplot,'Scatterplot_Task_3_V-
A_Static.jpg')                        %uncomment to save figure

%% -------------------------  TASK 2 subquestions 5+6  ----------------------------
------ %%

% PDL2 = Pupil Diameter Right, Task 2
% VL2  = Gaze Y Right, Task 2
% AL2  = Refraction Right, Task 2
% PDR2 = Pupil Diameter Right, Task 2
% VR2  = Gaze Y Right, Task 2
% AR2  = Refraction Right, Task 2

% Inladen variabelen
DataPDL2 = DataTask2{2,3};
DataVL2  = DataTask2{2,5};
DataAL2  = DataTask2{2,6};

DataPDR2 = DataTask2{2,9};
DataVR2  = DataTask2{2,11};
DataAR2  = DataTask2{2,12};

k = 1;
for nMultiplications = 1:9
    PDL2          = squeeze(DataPDL2(:,nMultiplications,:));
    VL2           = squeeze(DataVL2(:,nMultiplications,:));
    AL2           = squeeze(DataAL2(:,nMultiplications,:));

    PDR2          = squeeze(DataPDR2(:,nMultiplications,:));
    VR2           = squeeze(DataVR2(:,nMultiplications,:));
    AR2           = squeeze(DataAR2(:,nMultiplications,:));

    PDL2_Row{1,k} = PDL2;
    VL2_Row{1,k}  = VL2;
    AL2_Row{1,k}  = AL2;
    PDR2_Row{1,k} = PDR2;
    VR2_Row{1,k}  = VR2;
    AR2_Row{1,k}  = AR2;
    k = k + 1;
end


%% (SUBQUESTIONS 5+6)
% Subquestion (5+6): comparing easy, medium and hard multiplications with eachother
```

```
Indices56 = [1:3;
             4:6;
             7:9];

for n = 1:3                                                                % links
en recht gemiddeld, van de 3 moeilijkheidsgraden
    PDL2_Temp56            = DataPDL2(:,Indices56(n,:),:);
    PDR2_Temp56            = DataPDR2(:,Indices56(n,:),:);
    PDL2_MeanMatrix56{n,1} = squeeze(nanmean(PDL2_Temp56,2));
    PDR2_MeanMatrix56{n,1} = squeeze(nanmean(PDR2_Temp56,2));
    PD2_MeanMatrixGem56    = (PDL2_MeanMatrix56{n,1} + PDR2_MeanMatrix56{n,1})/2;
    PD2_MeanMatrix56{n,1}  = PD2_MeanMatrixGem56;
end

for n = 1:3                                                                % links
en recht gemiddeld, van de 3 moeilijkheidsgraden
    VL2_Temp56            = DataVL2(:,Indices56(n,:),:);
    VR2_Temp56            = DataVR2(:,Indices56(n,:),:);
    VL2_MeanMatrix56{n,1} = squeeze(nanmean(VL2_Temp56,2));
    VR2_MeanMatrix56{n,1} = squeeze(nanmean(VR2_Temp56,2));
    V2_MeanMatrixGem56    = (VL2_MeanMatrix56{n,1} + VR2_MeanMatrix56{n,1})/2;
    V2_MeanMatrix56{n,1}  = V2_MeanMatrixGem56;
end

for n = 1:3                                                                %links
en recht gemiddeld, van de 3 moeilijkheidsgraden
    AL2_Temp56            = DataAL2(:,Indices56(n,:),:);
    AR2_Temp56            = DataAR2(:,Indices56(n,:),:);
    AL2_MeanMatrix56{n,1} = squeeze(nanmean(AL2_Temp56,2));
    AR2_MeanMatrix56{n,1} = squeeze(nanmean(AR2_Temp56,2));
    A2_MeanMatrixGem56    = (AL2_MeanMatrix56{n,1} + AR2_MeanMatrix56{n,1})/2;
    A2_MeanMatrix56{n,1}  = A2_MeanMatrixGem56;
end


% PD2C = Pupil Diameter Change in Percentage
% V2C = Vergence, Gaze Y Change in Percentage
% A2C = Accommodation, refraction Change in Percentage
PD2C_Level56 = NaN(35,3);
for nLevel56 = 1:3
    for nParticipants = 1 : 35
        MeanControlSlidePD56 =
nanmean(PD2_MeanMatrix56{nLevel56,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoPD56         =
max(PD2_MeanMatrix56{nLevel56,1}(50*10.5:50*24.5,nParticipants));
        PD2C_Level56X       = (MaxVideoPD56 -
MeanControlSlidePD56)/MeanControlSlidePD56;
        PD2C_Level56(nParticipants,nLevel56) = PD2C_Level56X*100;             %
kolom 1 is makkelijk, kolom 2 is middel, kolom 3 is moeilijke sommen.
    end
end

V2C_Level56 = NaN(35,3);
for nLevel56 = 1:3
    for nParticipants = 1 : 35
        MeanControlSlideV56 =
nanmean(V2_MeanMatrix56{nLevel56,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoV56         =
max(V2_MeanMatrix56{nLevel56,1}(50*10.5:50*24.5,nParticipants));
        V2C_Level56X       = (MaxVideoV56 -
MeanControlSlideV56)/MeanControlSlideV56;
        V2C_Level56(nParticipants,nLevel56) = V2C_Level56X*100;               %
kolom 1 is makkelijk, kolom 2 is middel, kolom 3 is moeilijke sommen.
    end
end

A2C_Level56 = NaN(35,3);
```

```
for nLevel56 = 1:3
    for nParticipants = 1 : 35
        MeanControlSlideA56 =
nanmean(A2_MeanMatrix56{nLevel56,1}(50*0.5:50*9.5,nParticipants));
        MaxVideoA56         =
max(A2_MeanMatrix56{nLevel56,1}(50*10.5:50*24.5,nParticipants));
        A2C_Level56X       = (MaxVideoA56 -
MeanControlSlideA56)/MeanControlSlideA56;
        A2C_Level56(nParticipants,nLevel56) =
A2C_Level56X*100;                           % kolom 1 is makkelijk, kolom 2 is
middel, kolom 3 is moeilijke sommen.
    end
end



%% MAKKELIJK PD2 <-> A2 (SUBQUESTION 5+6)
% PD2 <-> A2
fitA56 = PD2C_Level56(:,1);     % X-as
fitB56 = A2C_Level56(:,1);      % Y-as
for n=1:35
    if fitA56(n,1)<2.1868 || fitA56(n,1)>11.4794
            fitA56(n,1)  = NaN;
            fitB56(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB56(n,1)<-93.8012 || fitB56(n,1)>34.5726
            fitB56(n,1)  = NaN;
            fitA56(n,1) =  NaN
    end
end
isnan(fitA56(:,1))
VectorNaN = isnan(fitA56(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA56(isnan(fitA56))=[];
fitB56(isnan(fitB56))=[];
scatter(fitA56,fitB56,35,'b','*')
P = polyfit(fitA56,fitB56,1);
FitLine = P(1)*fitA56+P(2); hold on;
Scatterplot = plot(fitA56,FitLine,'color',colors(2,:),'LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 2: Easy multiplications');
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1)
%saveas(Scatterplot,'Scatterplot_Task_2_PD-
A_easy_multiplications.jpg')     %uncomment to save figure

%% Middel moeilijk PD2 <-> A2 (SUBQUESTION 5+6)
% PD2 <-> A2
fitA56 = PD2C_Level56(:,2);     % X-as
fitB56 = A2C_Level56(:,2);      % Y-as
for n=1:35
    if fitA56(n,1)<3.4852 || fitA56(n,1)>11.8882
            fitA56(n,1)  = NaN;
            fitB56(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB56(n,1)<-135.4164 || fitB56(n,1)>89.3018
            fitB56(n,1)  = NaN;
            fitA56(n,1) =  NaN
```

```
    end
end
isnan(fitA56(:,1))
VectorNaN = isnan(fitA56(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA56(isnan(fitA56))=[];
fitB56(isnan(fitB56))=[];
scatter(fitA56,fitB56,35,'b','*')
P = polyfit(fitA56,fitB56,1);
FitLine = P(1)*fitA56+P(2); hold on;
Scatterplot = plot(fitA56,FitLine,'color',colors(15,:),'LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 2: Middle hard multiplications');
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1)
%saveas(Scatterplot,'Scatterplot_Task_2_PD-
A_middle_hard_multiplications.jpg')                         %uncomment to save
figure


%% Moeilijk PD2 <-> A2 (SUBQUESTION 5+6)
% PD2 <-> A2
fitA56 = PD2C_Level56(:,3);     % X-as
fitB56 = A2C_Level56(:,3);      % Y-as
for n=1:35
    if fitA56(n,1)<2.3562 || fitA56(n,1)>13.3112
            fitA56(n,1)  = NaN;
            fitB56(n,1)  = NaN;
    end
end
for n=1:35
    if  fitB56(n,1)<-2.0325e+03|| fitB56(n,1)>1.4101e+03
            fitB56(n,1)  = NaN;
            fitA56(n,1) =  NaN;
    end
end
isnan(fitA56(:,1))
VectorNaN = isnan(fitA56(:,1))
TotalNan = sum(VectorNaN)
participants = 35-TotalNan;
fitA56(isnan(fitA56))=[];
fitB56(isnan(fitB56))=[];
scatter(fitA56,fitB56,35,'b','*')
P = polyfit(fitA56,fitB56,1);
FitLine = P(1)*fitA56+P(2); hold on;
Scatterplot = plot(fitA56,FitLine,'color',colors(8,:),'LineWidth',3); hold on;
slope = P(1);
title('Scatterplot of Task 2: Hard multiplications');
xlabel('Pupil Diameter Change [%]');
ylabel('Refraction Change [%]');
grid on; grid minor;
set(gcf,'Position',get(0,'Screensize'));
legendInfo1 = sprintf('Fitted curve with slope = {%0.1f}', (slope));
legendinfo2 = sprintf('total participants = {%1.f}', (participants));
legend(legendinfo2,legendInfo1)
%saveas(Scatterplot,'Scatterplot_Task_2_PD-
A_hard_multiplications.jpg')     %uncomment to save figure

%% ---------- Cross Correlation // Task 1 (low tones) ----------------- %%
close all; clc;
PDL1_1_xcorr = PDL1_1_Plot(:,1);
PDR1_1_xcorr = PDR1_1_Plot(:,1);
```

```
PDL1_1_xcorr(isnan(PDL1_1_xcorr)) = [];
PDR1_1_xcorr(isnan(PDR1_1_xcorr)) = [];
[xcorr_c_PD1_1,xcorr_lags_PD1_1] = xcorr(PDL1_1_xcorr,PDR1_1_xcorr);
figPD1_1 = plot(xcorr_lags_PD1_1,xcorr_c_PD1_1/max(xcorr_c_PD1_1)); hold on;
title('Cross-Correlation Task 1 (low tones): Pupil Diameter')
% saveas(figPD1_1,'Cross-Correlation Task 1 (low tones) Pupil Diameter.jpg')
figure
VL1_1_xcorr = VL1_1_Plot(:,1);
VR1_1_xcorr = VR1_1_Plot(:,1);
VL1_1_xcorr(isnan(VL1_1_xcorr)) = [];
VR1_1_xcorr(isnan(VR1_1_xcorr)) = [];
[xcorr_c_V1_1,xcorr_lags_V1_1] = xcorr(VL1_1_xcorr,VR1_1_xcorr);
figV1_1 = plot(xcorr_lags_V1_1,xcorr_c_V1_1/max(xcorr_c_V1_1)); hold on;
title('Cross-Correlation Task 1 (low tones): Vergence, Gaze X')
% saveas(figV1_1,'Cross-Correlation Task 1 (low tones) Vergence, Gaze X.jpg')
figure
AL1_1_xcorr = AL1_1_Plot(:,1);
AR1_1_xcorr = AR1_1_Plot(:,1);
AL1_1_xcorr(isnan(AL1_1_xcorr)) = [];
AR1_1_xcorr(isnan(AR1_1_xcorr)) = [];
[xcorr_c_A1_1,xcorr_lags_A1_1] = xcorr(AL1_1_xcorr,AR1_1_xcorr);
figA1_1 = plot(xcorr_lags_A1_1,xcorr_c_A1_1/max(xcorr_c_A1_1)); hold on;
title('Cross-Correlation Task 1 (low tones): Accommodation, Refraction')
% saveas(figA1_1,'Cross-Correlation Task 1 (low tones) Accommodation,
Refraction.jpg')
%% ----------- Cross Correlation // Task 1 (high tones) ---------------- %%
close all; clc;
PDL1_2_xcorr = PDL1_2_Plot(:,1);
PDR1_2_xcorr = PDR1_2_Plot(:,1);
PDL1_2_xcorr(isnan(PDL1_2_xcorr)) = [];
PDR1_2_xcorr(isnan(PDR1_2_xcorr)) = [];
[xcorr_c_PD1_2,xcorr_lags_PD1_2] = xcorr(PDL1_2_xcorr,PDR1_2_xcorr);
figPD1_2 = plot(xcorr_lags_PD1_2,xcorr_c_PD1_2/max(xcorr_c_PD1_2)); hold on;
title('Cross-Correlation Task 1 (high tones): Pupil Diameter')
% saveas(figPD1_2,'Cross-Correlation Task 1 (high tones) Pupil Diameter.jpg')
figure
VL1_2_xcorr = VL1_2_Plot(:,1);
VR1_2_xcorr = VR1_2_Plot(:,1);
VL1_2_xcorr(isnan(VL1_2_xcorr)) = [];
VR1_2_xcorr(isnan(VR1_2_xcorr)) = [];
[xcorr_c_V1_2,xcorr_lags_V1_2] = xcorr(VL1_2_xcorr,VR1_2_xcorr);
figV1_2 = plot(xcorr_lags_V1_2,xcorr_c_V1_2/max(xcorr_c_V1_2)); hold on;
title('Cross-Correlation Task 1 (high tones): Vergence, Gaze X')
% saveas(figV1_2,'Cross-Correlation Task 1 (high tones) Vergence, Gaze X.jpg')
figure
AL1_2_xcorr = AL1_2_Plot(:,1);
AR1_2_xcorr = AR1_2_Plot(:,1);
AL1_2_xcorr(isnan(AL1_2_xcorr)) = [];
AR1_2_xcorr(isnan(AR1_2_xcorr)) = [];
[xcorr_c_A1_2,xcorr_lags_A1_2] = xcorr(AL1_2_xcorr,AR1_2_xcorr);
figA1_2 = plot(xcorr_lags_A1_2,xcorr_c_A1_2/max(xcorr_c_A1_2)); hold on;
title('Cross-Correlation Task 1 (high tones): Accommodation, Refraction')
% saveas(figA1_2,'Cross-Correlation Task 1 (high tones) Accommodation,
Refraction.jpg')

%% ------------------ Cross-Correlation // Task 2 ---------------------- %%
close all; clc;
PDL2_xcorr = PDL2_Plot(:,1);
PDR2_xcorr = PDR2_Plot(:,1);
PDL2_xcorr(isnan(PDL2_xcorr)) = [];
PDR2_xcorr(isnan(PDR2_xcorr)) = [];
[xcorr_c_PD2,xcorr_lags_PD2] = xcorr(PDL2_xcorr,PDR2_xcorr);
figPD2 = plot(xcorr_lags_PD2,xcorr_c_PD2/max(xcorr_c_PD2)); hold on;
title('Cross-Correlation Task 2: Pupil Diameter')
% saveas(figPD2,'Cross-Correlation Task 2 Pupil Diameter.jpg')
figure
AL2_xcorr = AL2_Plot(:,1);
AR2_xcorr = AR2_Plot(:,1);
```

```
AL2_xcorr(isnan(AL2_xcorr)) = [];
AR2_xcorr(isnan(AR2_xcorr)) = [];
[xcorr_c_A2,xcorr_lags_A2] = xcorr(AL2_xcorr,AR2_xcorr);
figA2 = plot(xcorr_lags_A2,xcorr_c_A2/max(xcorr_c_A2)); hold on;
title('Cross-Correlation Task 2: Accommodation, Refraction')
% saveas(figA2,'Cross-Correlation Task 2 Accommodation, Refraction.jpg')

%% ------------------ Cross-Correlation // Task 3 ---------------------- %%
close all; clc;
PDL3_xcorr = PDL3_Plot(:,1);
PDR3_xcorr = PDR3_Plot(:,1);
PDL3_xcorr(isnan(PDL3_xcorr)) = [];
PDR3_xcorr(isnan(PDR3_xcorr)) = [];
[xcorr_c_PD3,xcorr_lags_PD3] = xcorr(PDL3_xcorr,PDR3_xcorr);
figPD3 = plot(xcorr_lags_PD3,xcorr_c_PD3/max(xcorr_c_PD3)); hold on;
title('Cross-Correlation Task 3: Pupil Diameter')
% saveas(figPD3,'Cross-Correlation Task 3 Pupil Diameter.jpg')
figure
VL3_xcorr = VL3_Plot(:,1);
VR3_xcorr = VR3_Plot(:,1);
VL3_xcorr(isnan(VL3_xcorr)) = [];
VR3_xcorr(isnan(VR3_xcorr)) = [];
[xcorr_c_V3,xcorr_lags_V3] = xcorr(VL3_xcorr,VR3_xcorr);
figV3 = plot(xcorr_lags_V3,xcorr_c_V3/max(xcorr_c_V3)); hold on;
title('Cross-Correlation Task 3: Vergence, Gaze Y')
% saveas(figV3,'Cross-Correlation Task 3 Vergence, Gaze Y.jpg')
figure
AL3_xcorr = AL3_Plot(:,1);
AR3_xcorr = AR3_Plot(:,1);
AL3_xcorr(isnan(AL3_xcorr)) = [];
AR3_xcorr(isnan(AR3_xcorr)) = [];
[xcorr_c_A3,xcorr_lags_A3] = xcorr(AL3_xcorr,AR3_xcorr);
figA3 = plot(xcorr_lags_A3,xcorr_c_A3/max(xcorr_c_A3)); hold on;
title('Cross-Correlation Task 3: Accommodation')
% saveas(figA3,'Cross-Correlation Task 3 Accommodation, Refraction.jpg')
```

## C.6 Subplotgeneratorpaper.m

See next page.

```
%% Replicatie BEP-groep - Powerref
% Matlab script build by Lars Kooijman to analyse data
% Edited by Stefan Jansen, Julia Russell and Tamir Themans


clear all; %#ok<*CLALL>
close all;clc;
opengl('save','software')

PilotData       = 0;
ExperimentData  = 1;


figure
sub1 = subplot(3,1,1); hold on;
sub2 = subplot(3,1,2);
sub3 = subplot(3,1,3);
Dimensions_3_Subplots = [sub1.Position;sub2.Position;sub3.Position];
%hold off;
%close all;

figure
sub1 = subplot(2,1,1); hold on;
sub2 = subplot(2,1,2);
Dimensions_2_Subplots = [sub1.Position;sub2.Position];
hold off;
close all;
clear sub1 sub2 sub3
%% Initialization
rootFolder   = pwd;
if PilotData == 1
    dataFolder = strcat(rootFolder,'\FiltData\Pilot');
    cd(dataFolder);
    load FiltDataSoundsLow
    load FiltDataSoundsHigh
    load FiltDataMultiplications
    load FiltDataVideos
%     fileNames  = dir('*.mat');

elseif ExperimentData == 1
    dataFolder = strcat(rootFolder,'\FiltData\Experiment');
    cd(dataFolder);
    load FiltDataSoundsLow
    load FiltDataSoundsHigh
    load FiltDataMultiplications
    load FiltDataVideos
end

%% Variables

% 1.  Pupil Size X Left
% 2.  Pupil Size Y Left
% 3.  Pupil Diameter Left    <--    PDL
% 4.  Gaze X Left            <--    VL (X)
% 5.  Gaze Y Left            <--    VL (Y)
% 6.  Refraction Left        <--    AL
%
% 7.  Pupil Size X Right
% 8.  Pupil Size Y Right
% 9.  Pupil Diameter Right   <--    PDR
% 10. Gaze X Right           <--    VR (X)
% 11. Gaze Y Right           <--    VR (Y)
% 12. Refraction Right       <--    AR


% Task 1: PDL, VL (X), AL
%         PDR, VR (X), AR
% Task 2: PDL, AL
```

```
%         PDR, AR
% Task 3: PDL, VL (Y), AL
%         PDR, VR (Y), AR

%% Colors for plots
colors = [204 255 153 ;       % 1 lichtgroen
          153 255 51;         % 2 middelgroen
          102 204 0;          % 3 donkergroen
          153 153 255;        % 4 lichtblauw
          51  51  255;        % 5 middelblauw
          0   0   204;        % 6 donkerblauw
          255 153 153;        % 7 lichtrood
          255 51  51;         % 8 middelrood
          204 0   0;          % 9 donkerrood
          255 102 178;        % 10 lichtroze
          255 0   127;        % 11 middelroze
          153 0   76 ;        % 12 donkerroze
          153 0   76;         % 13 donkerpaars
          0   255 255;        % 14 turquiose
          0   128 255;        % 15 donkerturquiose
          0   0   255;        % 16 nog donderder blauw
          255 128 0;          % 17 oranje
          255 255 0;          % 18 geel
          153 0   153;        % 19 paars
          255 51  255]/255;   % 20 knalroze




%% --------------------  TASK 1 (low&high) ----------------------------------- %%

% PDL1 = Pupil Diameter Left, Task 1
% VL1  = Gaze X Left, Task 1
% AL1  = Refraction Left, Task 1
% PDR1 = Pupil Diameter Right, Task 1
% VR1  = Gaze X Right, Task 1
% AR1  = Refraction Right, Task 1

% Inladen variabelen
DataVL1_2  = DataTask1_2{2,4}(:,1:3,:);                                   % Gaze X
- High Tones - Left
DataVR1_2  = DataTask1_2{2,10}(:,1:3,:);                                  % Gaze X
- High Tones - Right

DataAL1_2  = DataTask1_2{2,6}(:,1:3,:);                                   %
Refraction - High Tones -  Left
DataAR1_2  = DataTask1_2{2,12}(:,1:3,:);                                  %
Refraction - High Tones -  Right

DataVL1_1  = DataTask1_1{2,4}(:,1:3,:);                                   % Gaze X
- Low Tones - Left
DataVR1_1  = DataTask1_1{2,10}(:,1:3,:);                                  % Gaze X
- Low Tones - Right

DataAL1_1  = DataTask1_1{2,6}(:,1:3,:);                                   %
Refraction - Low Tones -  Left
DataAR1_1  = DataTask1_1{2,12}(:,1:3,:);                                  %
Refraction - Low Tones -  Right

% Mean Over Eyes
DataV1_2MeanEyes    = (DataVL1_2+DataVR1_2)/2;
DataA1_2MeanEyes    = (DataAL1_2+DataAR1_2)/2;
DataV1_1MeanEyes    = (DataVL1_1+DataVR1_1)/2;
DataA1_1MeanEyes    = (DataAL1_1+DataAR1_1)/2;

% Mean Over Repeated Tones (both eyes)
DataV1_2_MeanTones  = squeeze(nanmean(DataV1_2MeanEyes,2));               %mean 3
high tones Vergence
```

```
DataA1_2_MeanTones  = squeeze(nanmean(DataA1_2MeanEyes,2));                %mean 3
high tones Accommodation

DataV1_1_MeanTones  = squeeze(nanmean(DataV1_1MeanEyes,2));                %mean 3
low tones Vergence
DataA1_1_MeanTones  = squeeze(nanmean(DataA1_1MeanEyes,2));                %mean 3
low tones Accommodation

% Mean Over Participants (both eyes)
DataV1_2_MeanPar   = squeeze(nanmean(DataV1_2_MeanTones,2));               %all
participants together(1 value over time)
DataA1_2_MeanPar   = squeeze(nanmean(DataA1_2_MeanTones,2));               %all
participants together(1 value over time)

DataV1_1_MeanPar   = squeeze(nanmean(DataV1_1_MeanTones,2));               %all
participants together(1 value over time)
DataA1_1_MeanPar   = squeeze(nanmean(DataA1_1_MeanTones,2));               %all
participants together(1 value over time)

Time_Low      = 0:1/50:length(DataV1_1_MeanPar)/50 - 1/50;
Time_High     = 0:1/50:length(DataV1_2_MeanPar)/50 - 1/50;

fig= figure
sub1 = subplot(2,1,1);
plot(Time_Low ,DataA1_1_MeanPar,'LineWidth',2,'color',colors(14,:)); hold on
plot(Time_High,DataA1_2_MeanPar,'LineWidth',2,'color',colors(5,:));
xlim([0 10]);
title('Task 1: Low tones vs. High tones');
ylabel('Refraction [D]');
grid on;
grid minor;
legend('Low tones','High tones','Location','northeastoutside');

sub2 = subplot(2,1,2);
plot(Time_Low ,DataV1_1_MeanPar,'LineWidth',2,'color',colors(14,:)); hold on
plot(Time_High,DataV1_2_MeanPar,'LineWidth',2,'color',colors(5,:));
xlim([0 10]);
xlabel('Time [s]');
ylabel('Gaze (X) [Deg]');
grid on;
grid minor;
legend('Low tones','High tones','Location','northeastoutside');
set(gcf,'Position',[1 1 960 450])
StandardSizeSub1 = Dimensions_2_Subplots(1,:);
AdjustedSizeSub1 = StandardSizeSub1 - [0.048 0 -0.0  -0.0443];
StandardSizeSub2 = Dimensions_2_Subplots(2,:);
AdjustedSizeSub2 = StandardSizeSub2 - [0.048 0 -0.0  -0.0443];
sub1.Position    = AdjustedSizeSub1;
sub2.Position    = AdjustedSizeSub2;
hold off;

%saveas(fig,'Task1_Subplot_Low_tones_vs._High_Tones2.jpg')
% hold off

%% KIJKHOEK PLOT

% Mean over repeated tones (eyes seperate) [V]
DataVL1_2_Mean  = squeeze(nanmean(DataVL1_2,2));                             % low
tones
DataVR1_2_Mean  = squeeze(nanmean(DataVR1_2,2));

DataVL1_1_Mean  = squeeze(nanmean(DataVL1_1,2));                             % high
tones
DataVR1_1_Mean  = squeeze(nanmean(DataVR1_1,2));

% Mean Over Participants (eyes seperate) [V]
DataVL1_2_MP   = squeeze(nanmean(DataVL1_2_Mean,2));
DataVR1_2_MP   = squeeze(nanmean(DataVR1_2_Mean,2));
```

```
DataVL1_1_MP   = squeeze(nanmean(DataVL1_1_Mean,2));
DataVR1_1_MP   = squeeze(nanmean(DataVR1_1_Mean,2));

DataV1_2_vergence = abs(-DataVL1_2_MP + DataVR1_2_MP);
DataV1_1_vergence = abs(-DataVL1_1_MP + DataVR1_1_MP);


fig= figure
sub1 = subplot(3,1,1);
plot(Time_High,DataVL1_2_MP,'LineWidth',2,'color',colors(14,:)); hold on   % high
tone, left an right
plot(Time_High,DataVR1_2_MP,'LineWidth',2,'color',colors(5,:));
xlim([0 10]);
title('Task 1: low tones vs. high tones');
ylabel('Gaze (X) [Deg]');
grid on;
grid minor;
legend('High tones, Gaze X left','High tones, Gaze X
right','Location','northeastoutside');

sub2 = subplot(3,1,2);
plot(Time_Low,DataVL1_1_MP,'LineWidth',2,'color',colors(14,:)); hold on  % Low
tone, left an right
plot(Time_Low,DataVR1_1_MP,'LineWidth',2,'color',colors(5,:));
xlim([0 10]);
ylabel('Gaze (X) [Deg]');
grid on;
grid minor;
legend('Low tones, Gaze X left','Low tones, Gaze X
right','Location','northeastoutside')

sub3 = subplot(3,1,3);
plot(Time_High,DataV1_2_vergence,'LineWidth',2,'color',colors(14,:)); hold
on  %  both vergences, high and low tones
plot(Time_Low ,DataV1_1_vergence,'LineWidth',2,'color',colors(5,:));
xlim([0 10]);
xlabel('Time [s]');
ylabel('Vergence [Deg]');
grid on;
grid minor;
legend('High tones, Vergence','Low tones, Vergence','Location','northeastoutside')
set(gcf,'Position',[1 1 960 450])
StandardSizeSub1 = Dimensions_3_Subplots(1,:);
AdjustedSizeSub1 = StandardSizeSub1 - [0.048 0 0.05  -0.0443];
StandardSizeSub2 = Dimensions_3_Subplots(2,:);
AdjustedSizeSub2 = StandardSizeSub2 - [0.048 0 0.05  -0.0443];
StandardSizeSub3 = Dimensions_3_Subplots(3,:);
AdjustedSizeSub3 = StandardSizeSub3 - [0.048 0 0.05  -0.0443];
sub1.Position    = AdjustedSizeSub1;
sub2.Position    = AdjustedSizeSub2;
sub3.Position    = AdjustedSizeSub3;
hold off;



%saveas(fig,'Task_1_Subplot_refraction_High_Low_tones_Vergence2.jpg')

%% -------------------------  TASK 2  ---------------------------------- %%

% PDL2 = Pupil Diameter Left, Task 2
% VL2  = Gaze Y Left, Task 2
% AL2  = Refraction Left, Task 2
% PDR2 = Pupil Diameter Right, Task 2
% VR2  = Gaze Y Right, Task 2
% AR2  = Refraction Right, Task 2

%% EASY, AVERAGE, DIFFICULT
```

```
DataPDL2 = DataTask2{2,3};
DataAL2 = DataTask2{2,6};

DataPDR2 = DataTask2{2,9};
DataAR2 = DataTask2{2,12};
k = 1;
for nMultiplications = 1:9
    PDL2          = squeeze(DataPDL2(:,nMultiplications,:));
    AL2           = squeeze(DataAL2(:,nMultiplications,:));

    PDR2          = squeeze(DataPDR2(:,nMultiplications,:));
    AR2           = squeeze(DataAR2(:,nMultiplications,:));

    PDL2_Row{1,k} = PDL2;
    AL2_Row{1,k}  = AL2;
    PDR2_Row{1,k} = PDR2;
    AR2_Row{1,k}  = AR2;
    k = k + 1;
end


Indices = [1:3;
           4:6;
           7:9];

for n = 1:3
    PDL2_Temp            = DataPDL2(:,Indices(n,:),:);
    PDR2_Temp            = DataPDR2(:,Indices(n,:),:);
    PDL2_MeanMatrix{n,1} = squeeze(nanmean(PDL2_Temp,2));
    PDR2_MeanMatrix{n,1} = squeeze(nanmean(PDR2_Temp,2));
    PD2_MeanMatrixGem    = (PDL2_MeanMatrix{n,1} + PDR2_MeanMatrix{n,1})/2;
    PD2_MeanMatrix{n,1}  = PD2_MeanMatrixGem;
end

for n = 1:3
    AL2_Temp            = DataAL2(:,Indices(n,:),:);
    AR2_Temp            = DataAR2(:,Indices(n,:),:);
    AL2_MeanMatrix{n,1} = squeeze(nanmean(AL2_Temp,2));
    AR2_MeanMatrix{n,1} = squeeze(nanmean(AR2_Temp,2));
    A2_MeanMatrixGem    = (AL2_MeanMatrix{n,1} + AR2_MeanMatrix{n,1})/2;
    A2_MeanMatrix{n,1}  = A2_MeanMatrixGem;
end

PD2_easy = nanmean(PD2_MeanMatrix{1,1},2);
PD2_mid  = nanmean(PD2_MeanMatrix{2,1},2);
PD2_hard = nanmean(PD2_MeanMatrix{3,1},2);
A2_easy  = nanmean(A2_MeanMatrix{1,1},2);
A2_mid   = nanmean(A2_MeanMatrix{2,1},2);
A2_hard  = nanmean(A2_MeanMatrix{3,1},2);

PD2_Time_easy  = 0:1/50:length(PD2_easy)/50 - 1/50;                          %
bepaal lengte tijdsvector PD2, easy
PD2_Time_mid   = 0:1/50:length(PD2_mid)/50 - 1/50;                           %
bepaal lengte tijdsvector PD2, mid
PD2_Time_hard  = 0:1/50:length(PD2_hard)/50 - 1/50;                          %
bepaal lengte tijdsvector PD2, hard
A2_Time_easy   = 0:1/50:length(A2_easy)/50 - 1/50;                           %
bepaal lengte tijdsvector A2, easy
A2_Time_mid    = 0:1/50:length(A2_mid)/50 - 1/50;                            %
bepaal lengte tijdsvector A2, mid
A2_Time_hard   = 0:1/50:length(A2_hard)/50 - 1/50;                           %
bepaal lengte tijdsvector A2, hard


figure
sub1= subplot(2,1,1)
fig = plot(PD2_Time_easy,PD2_easy,'LineWidth',2,'color',colors(2,:)); hold on; %
plot PD2, easy
```

```
fig = plot(PD2_Time_mid,PD2_mid,'LineWidth',2,'color',colors(5,:)); hold on;   %
plot PD2, mid
fig = plot(PD2_Time_hard,PD2_hard,'LineWidth',2,'color',colors(8,:)); hold on; %
plot PD2, hard
fig= plot([10 10], [6 7], 'k');
xlim([0 25]); ylim([6.50 7.0]);
title('Task 2: Levels of difficulty');
ylabel('Pupil Diameter [mm]');
grid on; grid minor;
set(gca,'LooseInset',[0 0 0 0]);
legend('Easy multiplications','Average multiplications','Difficult
multiplications','Location','northeastoutside')
set(gcf,'Position',get(0,'Screensize'));

sub2= subplot(2,1,2)
fig = plot(A2_Time_easy,A2_easy,'LineWidth',2,'color',colors(2,:)); hold on; % plot
A2, easy
fig = plot(A2_Time_mid,A2_mid,'LineWidth',2,'color',colors(5,:)); hold on;   % plot
A2, mid
fig = plot(A2_Time_hard,A2_hard,'LineWidth',2,'color',colors(8,:)); hold on; % plot
A2, hard
fig= plot([10 10], [-0.7 -0.4], 'k');
xlim([0 25]); ylim([-0.65 -0.45]);
xlabel('Time [s]');
ylabel('Refraction [D]');
grid on; grid minor;
set(gca,'LooseInset',[0 0 0 0]);
legend('Easy multiplications','Average multiplications','Difficult
multiplications','Location','northeastoutside')
set(gcf,'Position',[1 1 960 450])
StandardSizeSub1 = Dimensions_2_Subplots(1,:);
AdjustedSizeSub1 = StandardSizeSub1 - [0.048 0 0.05  -0.0343];
StandardSizeSub2 = Dimensions_2_Subplots(2,:);
AdjustedSizeSub2 = StandardSizeSub2 - [0.048 0 0.05  -0.0343];
sub1.Position    = AdjustedSizeSub1;
sub2.Position    = AdjustedSizeSub2;

%saveas(fig,'subquestion_5_6_Subplot_Easy_Average_Difficult_Multiplications2.jpg')
hold off
%% -------------------------  TASK 3  ---------------------------------- %%

% PDL3 = Pupil Diameter Right, Task 3
% VL3  = Gaze Y Right, Task 3
% AL3  = Refraction Right, Task 3
% PDR3 = Pupil Diameter Right, Task 3
% VR3  = Gaze Y Right, Task 3
% AR3  = Refraction Right, Task 3

% Inladen variabelen
DataPDL3 = DataTask3{2,3};
DataVL3  = DataTask3{2,5};
DataAL3  = DataTask3{2,6};

DataPDR3 = DataTask3{2,9};
DataVR3  = DataTask3{2,11};
DataAR3  = DataTask3{2,12};

k = 1;
for nVideos = 1:24
    PDL3          = squeeze(DataPDL3(:,nVideos,:));
    VL3           = squeeze(DataVL3(:,nVideos,:));
    AL3           = squeeze(DataAL3(:,nVideos,:));

    PDR3          = squeeze(DataPDR3(:,nVideos,:));
    VR3           = squeeze(DataVR3(:,nVideos,:));
    AR3           = squeeze(DataAR3(:,nVideos,:));

    PDL3_Row{1,k} = PDL3;
```

```
    VL3_Row{1,k}  = VL3;
    AL3_Row{1,k}  = AL3;
    PDR3_Row{1,k} = PDR3;
    VR3_Row{1,k}  = VR3;
    AR3_Row{1,k}  = AR3;
    k = k + 1;
end

%% 4 LEVELS, PD3 // V3 // A3
Indices = [1:4:21;
           2:4:22;
           3:4:23;
           4:4:24];

for n = 1:4                                                              % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    PDL3_Temp            = DataPDL3(:,Indices(n,:),:);
    PDR3_Temp            = DataPDR3(:,Indices(n,:),:);
    PDL3_MeanMatrix{n,1} = squeeze(nanmean(PDL3_Temp,2));
    PDR3_MeanMatrix{n,1} = squeeze(nanmean(PDR3_Temp,2));
    PD3_MeanMatrixGem    = (PDL3_MeanMatrix{n,1} + PDR3_MeanMatrix{n,1})/2;
    PD3_MeanMatrix{n,1}  = PD3_MeanMatrixGem;
end

for n = 1:4                                                              % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    VL3_Temp            = DataVL3(:,Indices(n,:),:);
    VR3_Temp            = DataVR3(:,Indices(n,:),:);
    VL3_MeanMatrix{n,1} = squeeze(nanmean(VL3_Temp,2));
    VR3_MeanMatrix{n,1} = squeeze(nanmean(VR3_Temp,2));
    V3_MeanMatrixGem    = (VL3_MeanMatrix{n,1} + VR3_MeanMatrix{n,1})/2;
    V3_MeanMatrix{n,1}  = V3_MeanMatrixGem;
end

for n = 1:4                                                              % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    AL3_Temp            = DataAL3(:,Indices(n,:),:);
    AR3_Temp            = DataAR3(:,Indices(n,:),:);
    AL3_MeanMatrix{n,1} = squeeze(nanmean(AL3_Temp,2));
    AR3_MeanMatrix{n,1} = squeeze(nanmean(AR3_Temp,2));
    A3_MeanMatrixGem    = (AL3_MeanMatrix{n,1} + AR3_MeanMatrix{n,1})/2;
    A3_MeanMatrix{n,1}  = A3_MeanMatrixGem;
end

PD3_level_1 = nanmean(PD3_MeanMatrix{1,1},2);
PD3_level_2 = nanmean(PD3_MeanMatrix{2,1},2);
PD3_level_3 = nanmean(PD3_MeanMatrix{3,1},2);
PD3_level_4 = nanmean(PD3_MeanMatrix{4,1},2);
V3_level_1 = nanmean(V3_MeanMatrix{1,1},2);
V3_level_2 = nanmean(V3_MeanMatrix{2,1},2);
V3_level_3 = nanmean(V3_MeanMatrix{3,1},2);
V3_level_4 = nanmean(V3_MeanMatrix{4,1},2);
A3_level_1 = nanmean(A3_MeanMatrix{1,1},2);
A3_level_2 = nanmean(A3_MeanMatrix{2,1},2);
A3_level_3 = nanmean(A3_MeanMatrix{3,1},2);
A3_level_4 = nanmean(A3_MeanMatrix{4,1},2);

PD3_Time_level_1   = 0:1/50:length(PD3_level_1)/50 - 1/50;               %
bepaal lengte tijdsvector PD3, level X
PD3_Time_level_2   = 0:1/50:length(PD3_level_2)/50 - 1/50;               %
bepaal lengte tijdsvector PD3, level X
PD3_Time_level_3   = 0:1/50:length(PD3_level_3)/50 - 1/50;               %
bepaal lengte tijdsvector PD3, level X
PD3_Time_level_4   = 0:1/50:length(PD3_level_4)/50 - 1/50;               %
bepaal lengte tijdsvector PD3, level X
V3_Time_level_1   = 0:1/50:length(V3_level_1)/50 - 1/50;                 %
bepaal lengte tijdsvector V3, level X
```

```
V3_Time_level_2   = 0:1/50:length(V3_level_2)/50 - 1/50;                    %
bepaal lengte tijdsvector V3, level X
V3_Time_level_3   = 0:1/50:length(V3_level_3)/50 - 1/50;                    %
bepaal lengte tijdsvector V3, level X
V3_Time_level_4   = 0:1/50:length(V3_level_4)/50 - 1/50;                    %
bepaal lengte tijdsvector V3, level X
A3_Time_level_1   = 0:1/50:length(A3_level_1)/50 - 1/50;                    %
bepaal lengte tijdsvector A3, level X
A3_Time_level_2   = 0:1/50:length(A3_level_2)/50 - 1/50;                    %
bepaal lengte tijdsvector A3, level X
A3_Time_level_3   = 0:1/50:length(A3_level_3)/50 - 1/50;                    %
bepaal lengte tijdsvector A3, level X
A3_Time_level_4   = 0:1/50:length(A3_level_4)/50 - 1/50;                    %
bepaal lengte tijdsvector A3, level X

fig = figure;
sub1 = subplot(3,1,1);
plot(PD3_Time_level_1,PD3_level_1,'LineWidth',2,'color',colors(2,:)); hold on;  %
plot PD3, level X
plot(PD3_Time_level_2,PD3_level_2,'LineWidth',2,'color',colors(5,:)); hold on;  %
plot PD3, level X
plot(PD3_Time_level_3,PD3_level_3,'LineWidth',2,'color',colors(8,:)); hold on;  %
plot PD3, level X
plot(PD3_Time_level_4,PD3_level_4,'LineWidth',2,'color',colors(20,:)); hold on; %
plot PD3, level X
plot([10 10], [5 7], 'k');
xlim([0 16]); ylim([5.5 5.9]);
title('Task 3: All four levels of simplicity')
ylabel('Pupil Diameter [mm]');
grid on; grid minor;
legend('level 1: no road','level 2: simple road','level 3: average road','level 4:
realistic road', 'Location','northeastoutside')
set(gcf,'Position',get(0,'Screensize'));

sub2 = subplot(3,1,2);
plot(A3_Time_level_1,A3_level_1,'LineWidth',2,'color',colors(2,:)); hold on; % plot
A3, level X
plot(A3_Time_level_2,A3_level_2,'LineWidth',2,'color',colors(5,:)); hold on; % plot
A3, level X
plot(A3_Time_level_3,A3_level_3,'LineWidth',2,'color',colors(8,:)); hold on; % plot
A3, level X
plot(A3_Time_level_4,A3_level_4,'LineWidth',2,'color',colors(20,:)); hold on; %
plot A3, level X
plot([10 10], [-1 0], 'k');
xlim([0 16]); ylim([-0.5 -0.2]); yticks([-0.5 -0.4 -0.3 -0.2]);
ylabel('Refraction [D]');
grid on; grid minor;
legend('level 1: no road','level 2: simple road','level 3: average road','level 4:
realistic road','Location','northeastoutside')
set(gcf,'Position',get(0,'Screensize'));

sub3 = subplot(3,1,3);
plot(V3_Time_level_1,V3_level_1,'LineWidth',2,'color',colors(2,:)); hold on; % plot
V3, level X
plot(V3_Time_level_2,V3_level_2,'LineWidth',2,'color',colors(5,:)); hold on; % plot
V3, level X
plot(V3_Time_level_3,V3_level_3,'LineWidth',2,'color',colors(8,:)); hold on; % plot
V3, level X
plot(V3_Time_level_4,V3_level_4,'LineWidth',2,'color',colors(20,:)); hold on; %
plot V3, level X
plot([10 10], [-4 1.5], 'k');
xlim([0 16]); ylim([-4 2]);
xlabel('Time [s]');
ylabel('Gaze(Y) [Deg]');
grid on; grid minor;
set(gcf,'Position',[1 1 960 450])
legend('level 1: no road','level 2: simple road','level 3: average road','level 4:
realistic road', 'Location','northeastoutside')
```

```
StandardSizeSub1 = Dimensions_3_Subplots(1,:);
AdjustedSizeSub1 = StandardSizeSub1 - [0.048 0 0.04  -0.0343];
StandardSizeSub2 = Dimensions_3_Subplots(2,:);
AdjustedSizeSub2 = StandardSizeSub2 - [0.048 0 0.04  -0.0343];
StandardSizeSub3 = Dimensions_3_Subplots(3,:);
AdjustedSizeSub3 = StandardSizeSub3 - [0.048 0 0.04  -0.0343];
sub1.Position    = AdjustedSizeSub1;
sub2.Position    = AdjustedSizeSub2;
sub3.Position    = AdjustedSizeSub3;

saveas(fig,'subquestion_2_3_4_Subplot_All_Four_levels_of_simplicity2.jpg')
hold off

%% STATISCH EN DYNAMISCH, PD3 // A3 // V3

Indices = [5:8 13:16 21:24;
           1:4 9:12 17:20];

for n = 1:2                                                           % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    PDL3_Temp            = DataPDL3(:,Indices(n,:),:);
    PDR3_Temp            = DataPDR3(:,Indices(n,:),:);
    PDL3_MeanMatrix{n,1} = squeeze(nanmean(PDL3_Temp,2));
    PDR3_MeanMatrix{n,1} = squeeze(nanmean(PDR3_Temp,2));
    PD3_MeanMatrixGem    = (PDL3_MeanMatrix{n,1} + PDR3_MeanMatrix{n,1})/2;
    PD3_MeanMatrix{n,1}  = PD3_MeanMatrixGem;
end

for n = 1:2                                                           % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    VL3_Temp            = DataVL3(:,Indices(n,:),:);
    VR3_Temp            = DataVR3(:,Indices(n,:),:);
    VL3_MeanMatrix{n,1} = squeeze(nanmean(VL3_Temp,2));
    VR3_MeanMatrix{n,1} = squeeze(nanmean(VR3_Temp,2));
    V3_MeanMatrixGem    = (VL3_MeanMatrix{n,1} + VR3_MeanMatrix{n,1})/2;
    V3_MeanMatrix{n,1}  = V3_MeanMatrixGem;
end

for n = 1:2                                                           % links
en recht gemiddeld, van de 6 videos van level 1, enz. in totaal 4 levels
    AL3_Temp            = DataAL3(:,Indices(n,:),:);
    AR3_Temp            = DataAR3(:,Indices(n,:),:);
    AL3_MeanMatrix{n,1} = squeeze(nanmean(AL3_Temp,2));
    AR3_MeanMatrix{n,1} = squeeze(nanmean(AR3_Temp,2));
    A3_MeanMatrixGem    = (AL3_MeanMatrix{n,1} + AR3_MeanMatrix{n,1})/2;
    A3_MeanMatrix{n,1}  = A3_MeanMatrixGem;
end

PD3_static  = nanmean(PD3_MeanMatrix{1,1},2);
PD3_dynamic = nanmean(PD3_MeanMatrix{2,1},2);
V3_static   = nanmean(V3_MeanMatrix{1,1},2);
V3_dynamic  = nanmean(V3_MeanMatrix{2,1},2);
A3_static   = nanmean(A3_MeanMatrix{1,1},2);
A3_dynamic  = nanmean(A3_MeanMatrix{2,1},2);

PD3_Time_static   = 0:1/50:length(PD3_static)/50 - 1/50;                    %
bepaal lengte tijdsvector PD3, static
PD3_Time_dynamic  = 0:1/50:length(PD3_dynamic)/50 - 1/50;                   %
bepaal lengte tijdsvector PD3, dynamic
V3_Time_static    = 0:1/50:length(V3_static)/50 - 1/50;                     %
bepaal lengte tijdsvector V3, static
V3_Time_dynamic   = 0:1/50:length(V3_dynamic)/50 - 1/50;                    %
bepaal lengte tijdsvector V3, dynamic
A3_Time_static    = 0:1/50:length(A3_static)/50 - 1/50;                     %
bepaal lengte tijdsvector A3, static
A3_Time_dynamic   = 0:1/50:length(A3_dynamic)/50 - 1/50;                    %
bepaal lengte tijdsvector A3, dynamic
```

```
%% SUBPLOTS STATISCH VS. DYNAMISCH

1085 figure
sub1= subplot(3,1,1);
fig = plot(PD3_Time_static,PD3_static,'LineWidth',2,'color',colors(14,:)); hold on;
% plot PD3, static
fig = plot(PD3_Time_dynamic,PD3_dynamic,'LineWidth',2,'color',colors(5,:)); hold
on; % plot PD3, dynamic        % pupil diameter alle levels bij elkaar, van alle
loops, DYNAMISCH
plot([10 10], [5 7], 'k');
xlim([0 16]); ylim([5.55 5.85]);
title('Task 3: Static vs. Dynamic');
ylabel('Pupil Diameter [mm]');
grid on; grid minor;
legend('Static','Dynamic', 'Location','northeastoutside');

sub2= subplot(3,1,2);
fig = plot(A3_Time_static,A3_static,'LineWidth',2,'color',colors(14,:)); hold on; %
plot A3, static
fig = plot(A3_Time_dynamic,A3_dynamic,'LineWidth',2,'color',colors(5,:)); hold on;
% plot A3, dynamic        % dynamic accommodation dark blue plot alle levels bij
elkaar, van alle loops, STATISCH
fig= plot([10 10], [-1 0], 'k');
xlim([0 16]); ylim([-0.5 -0.199999]);
ylabel('Refraction [D]');
grid on; grid minor;
legend('Static ','Dynamic','Location','northeastoutside')

sub3= subplot(3,1,3);
fig = plot(V3_Time_static,V3_static,'LineWidth',2,'color',colors(14,:)); hold on; %
plot V3, static
fig = plot(V3_Time_dynamic,V3_dynamic,'LineWidth',2,'color',colors(5,:)); hold on;
% plot V3, dynamic       % dynamisch vergence dark blue
fig= plot([10 10], [-6 4], 'k');
xlim([0 16]); ylim([-4 2]);
xlabel('Time [s]');
ylabel('Gaze(Y) [Deg]');
grid on; grid minor;
set(gcf,'Position',[1 1 960 450])
legend('Static','Dynamic','Location','northeastoutside')
StandardSizeSub1 = Dimensions_3_Subplots(1,:);
AdjustedSizeSub1 = StandardSizeSub1 - [0.048 0 -0.01  -0.0343];
StandardSizeSub2 = Dimensions_3_Subplots(2,:);
AdjustedSizeSub2 = StandardSizeSub2 - [0.048 0 -0.01 -0.0343];
StandardSizeSub3 = Dimensions_3_Subplots(3,:);
AdjustedSizeSub3 = StandardSizeSub3 - [0.048 0 -0.01 -0.0343];
sub1.Position    = AdjustedSizeSub1;
sub2.Position    = AdjustedSizeSub2;
sub3.Position    = AdjustedSizeSub3;

%saveas(fig,'subquestion_1_Subplot_Static_vs_Dynamic2.jpg')

%hold off
```

## C.7 MovementVideosFinal.m

See next page.

```
%% Name Matlab Code
%  Creator : (Lars, 4-2019)
%  Checked and adapted by Julia Tamir en Stefan

clear all;%#ok<CLALL>
close all;
opengl hardware

Type = 2; % static == 1 dynamic == 2

%% 0. Name Section
% Create data

time         = 0:0.01:6;                                                  % Time
Vector
MotionPath1   = -1*(-535-exp((0.4*time-2.55).^2));                         % Motion
Marker
MotionPath2   = -1*(-535-exp((-0.4*time-0.15).^2));                        % Motion
Marker

samples       = 12 : 312;
StartSize     = 3.7*exp((0.0064*samples-2).^2);
func1         = [StartSize,fliplr(StartSize(1:end-1))];
StartSize2    = 3*exp((0.0064*samples-2).^2);
func2         = [StartSize2,fliplr(StartSize2(1:end-1))];

filename  = 'ballmovementvid8';
RingColor = [0 0 0];
FillColor2 = [127 127 127]/255;



%% 1. Name Section
% Create Environment

Background = imread('4_realistic_road_127.jpg');
image(Background); axis off


hFig = gcf;                                                                % get
the figure and axes handles
hAx  = gca;

set(hAx,'LooseInset',[0 0 0 0]);
set(hAx,'pos',[0 0 1 1]);
set(hAx,'Unit','normalized','Position',[0 0 1 1]);                         % set
the axes to full screen

set(hFig,'menubar','none','NumberTitle','off','WindowState','fullscreen')
set(hFig,'Unit','normalized','Position',[0 0 1 1]);                        % set
the axes to full screen
hold on;
v           = VideoWriter(filename,'MPEG-
4');                                      % Videowriter
v.FrameRate = 96;
open(v);                                                                  %
Videowriter
i = 1;
% Please explain what happens here in this loop. Other people need to
% understand what you are doing by reading your comments.
%first 30 frames were made unvisible; ball is entering script.
%frame 30-301(halfway) made by MotionPath1, when ball is returning(frame
%301-601), the ball follows MotionPath2

switch Type
    case 1
        %% Static Movies
        while i<=length(MotionPath1)
```

```
            if i>0 && i<=100
                h1(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func1(30),'MarkerFaceCo
lor',RingColor);
                h2(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func2(30),'MarkerFaceCo
lor',FillColor2);
            elseif i> 100 && i <= 200
                h1(i) =
plot(960,MotionPath1(115),'o','color',RingColor,'MarkerSize',func1(115),'MarkerFace
Color',RingColor);
                h2(i) =
plot(960,MotionPath1(115),'o','color',RingColor,'MarkerSize',func2(115),'MarkerFace
Color',FillColor2);
            elseif i> 200 && i <= 400
                h1(i) =
plot(960,MotionPath1(301),'o','color',RingColor,'MarkerSize',func1(301),'MarkerFace
Color',RingColor);
                h2(i) =
plot(960,MotionPath1(301),'o','color',RingColor,'MarkerSize',func2(301),'MarkerFace
Color',FillColor2);
            elseif i> 400 && i <= 500
                h1(i) =
plot(960,MotionPath1(115),'o','color',RingColor,'MarkerSize',func1(115),'MarkerFace
Color',RingColor);
                h2(i) =
plot(960,MotionPath1(115),'o','color',RingColor,'MarkerSize',func2(115),'MarkerFace
Color',FillColor2);
            elseif i> 500 && i <= 601
                h1(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func1(30),'MarkerFaceCo
lor',RingColor);
                h2(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func2(30),'MarkerFaceCo
lor',FillColor2);
            end
            if i >= 2
                set(h1(i-1),'Visible','off');
                set(h2(i-1),'Visible','off');
            end
        drawnow;
        i = i+1;
        frame = getframe(gcf);   %videowriter
        writeVideo(v,frame);     %videowriter
        end
        close(v)
        close all

    case 2
        %% Dynamic movies
        while i<=length(MotionPath1)
            if i>0 && i<=30
                h1(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func1(30),'MarkerFaceCo
lor',RingColor);
                h2(i) =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func2(30),'MarkerFaceCo
lor',FillColor2);
            elseif  i> 30 && i<=300.5
                h1(i) =
plot(960,MotionPath1(i),'o','color',RingColor,'MarkerSize',func1(i),'MarkerFaceColo
r',RingColor);
                h2(i) =
plot(960,MotionPath1(i),'o','color',RingColor,'MarkerSize',func2(i),'MarkerFaceColo
r',FillColor2);
                set(h1(i),'yData',MotionPath1(i));
                set(h2(i),'yData',MotionPath1(i));
            elseif i>300.5 && i<=572
```

```
                h1(i) =
plot(960,MotionPath2(i),'o','color',RingColor,'MarkerSize',func1(i),'MarkerFaceColo
r',RingColor);            % Here matlab tells you that the variable increases in
size every iteration. This is a bit sloppy coding.
                h2(i) =
plot(960,MotionPath2(i),'o','color',RingColor,'MarkerSize',func2(i),'MarkerFaceColo
r',FillColor2);
                set(h1(i),'yData',MotionPath2(i));
                set(h2(i),'yData',MotionPath2(i));

            else i > 572 &&  i<=601
                h1(i) =
plot(960,MotionPath2(572),'o','color',RingColor,'MarkerSize',func1(30),'MarkerFaceC
olor',RingColor);
                h2(i) =
plot(960,MotionPath2(572),'o','color',RingColor,'MarkerSize',func2(30),'MarkerFaceC
olor',FillColor2);
            end

            if i >= 2
                set(h1(i-1),'Visible','off');
                set(h2(i-1),'Visible','off');
            end
            drawnow;
            i = i+1;
            frame = getframe(gcf);   %videowriter
            writeVideo(v,frame);     %videowriter
        end
        close(v)
        close all
end

% %% Create Control Slide
% SaveThisBadBoy = figure;
% Background = imread('1_no_road_127.jpg');
% image(Background); axis off; hold on;
%
%
% hFig = gcf;                                                                % get
the figure and axes handles
% hAx  = gca;
%
% set(hAx,'LooseInset',[0 0 0 0]);
% set(hAx,'pos',[0 0 1 1]);
% set(hAx,'Unit','normalized','Position',[0 0 1 1]);                        % set
the axes to full screen
%
% set(hFig,'menubar','none','NumberTitle','off','WindowState','fullscreen')
% set(hFig,'Unit','normalized','Position',[0 0 1 1]);                       % set
the axes to full screen
% h1 =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func1(30),'MarkerFaceCo
lor',RingColor);
% h2 =
plot(960,MotionPath1(30),'o','color',RingColor,'MarkerSize',func2(30),'MarkerFaceCo
lor',FillColor2);
% saveas(SaveThisBadBoy,'Controlslide.jpg')
```

# D Appendix D

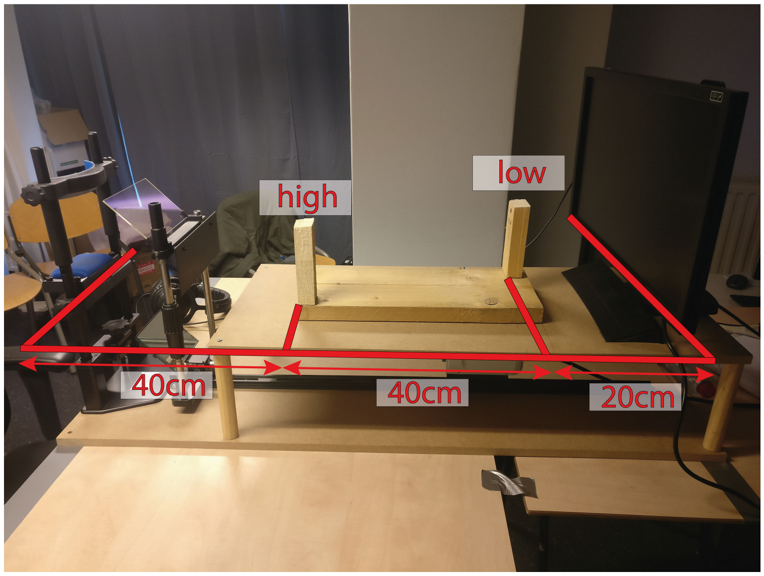


*Figure D.1: Setup PowerRef III with wooden object*

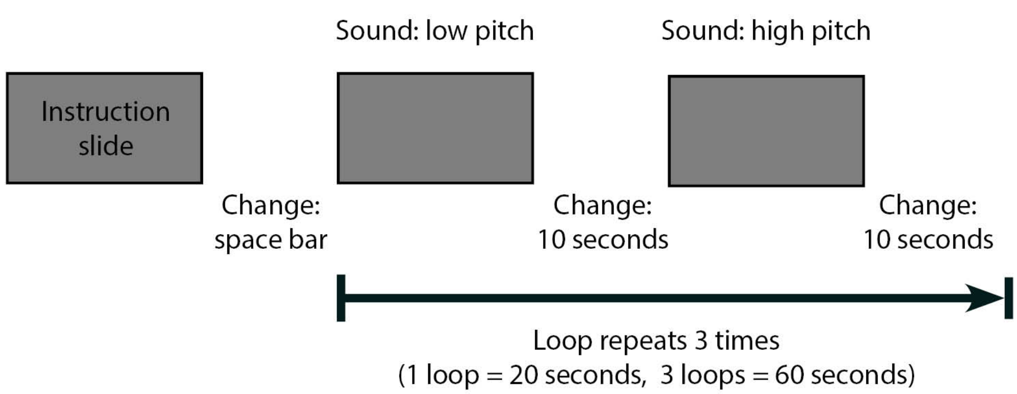


*Figure D.2: Detailed overview task 1*

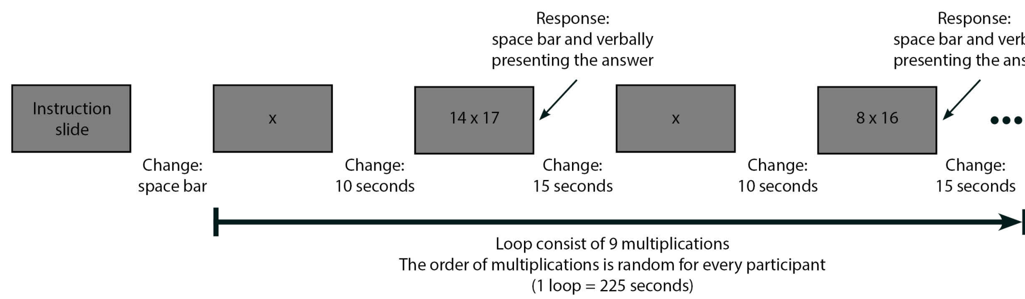


*Figure D.3: Detailed overview task 2*

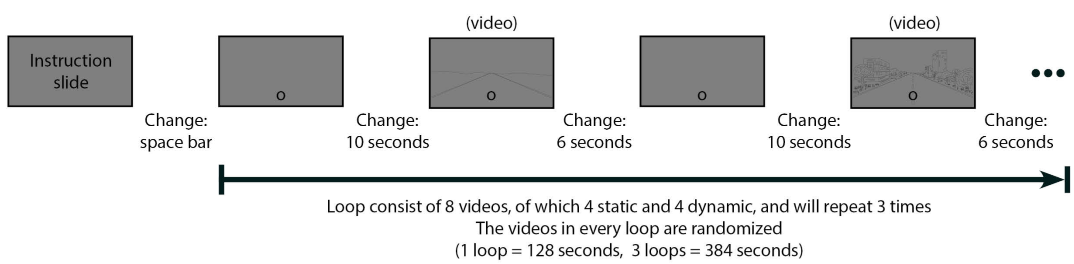


*Figure D.4: Detailed overview task 3*

# E Appendix E

*Figure E.1: Control slide task 1*

*Figure E.2: Control slide task 2*

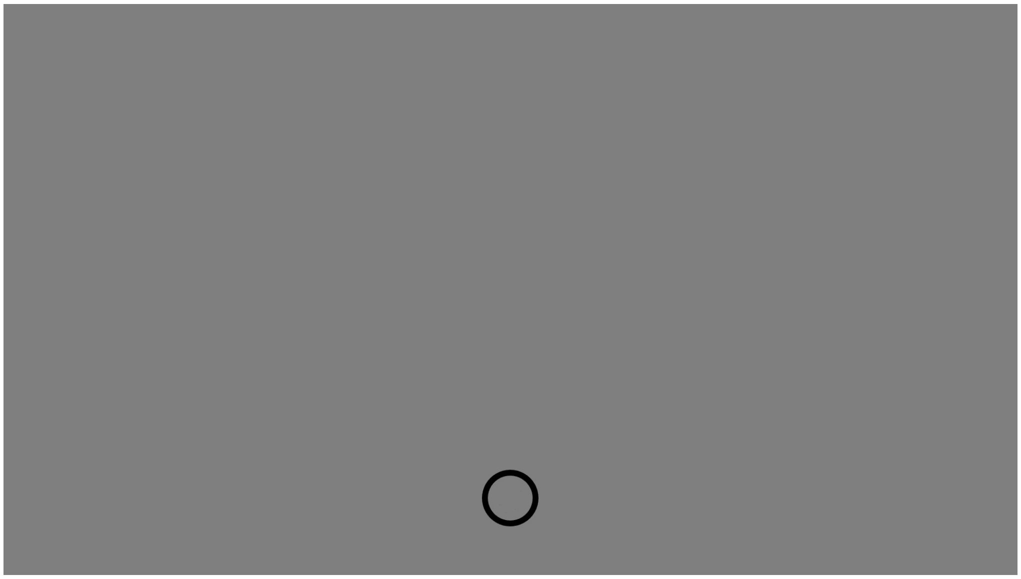

*Figure E.3: Control slide task 3*

# F Appendix F

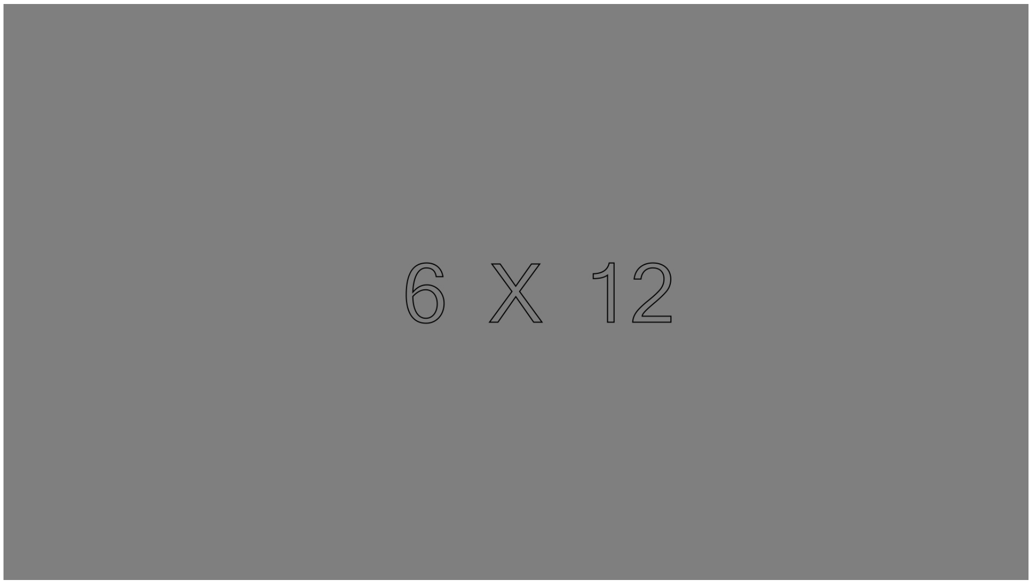


*Figure F.1: Multiplication slide of 6x12 during Task 2.*

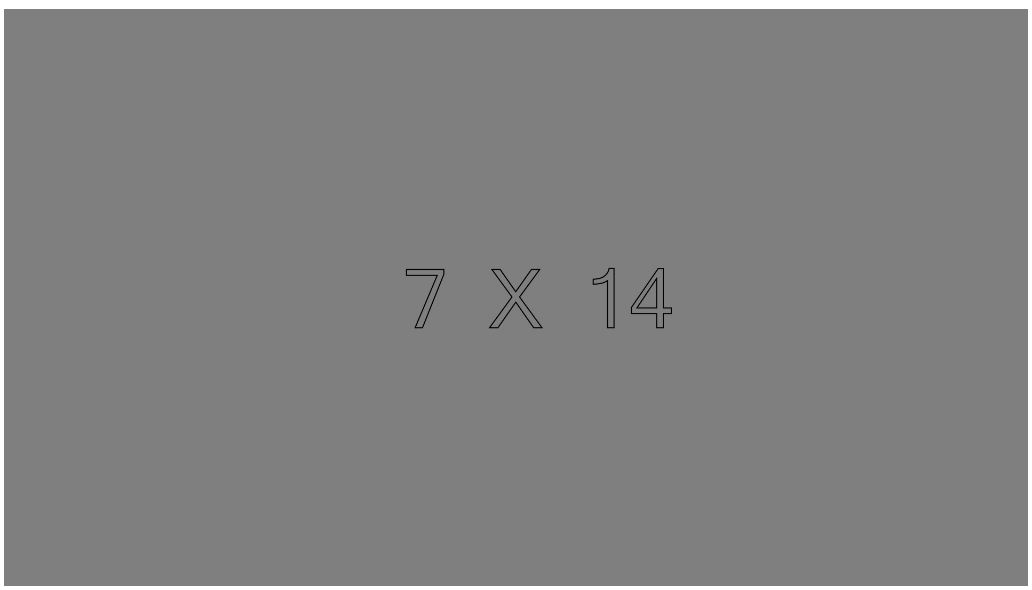


*Figure F.2: Multiplication slide of 7x14 during Task 2.*

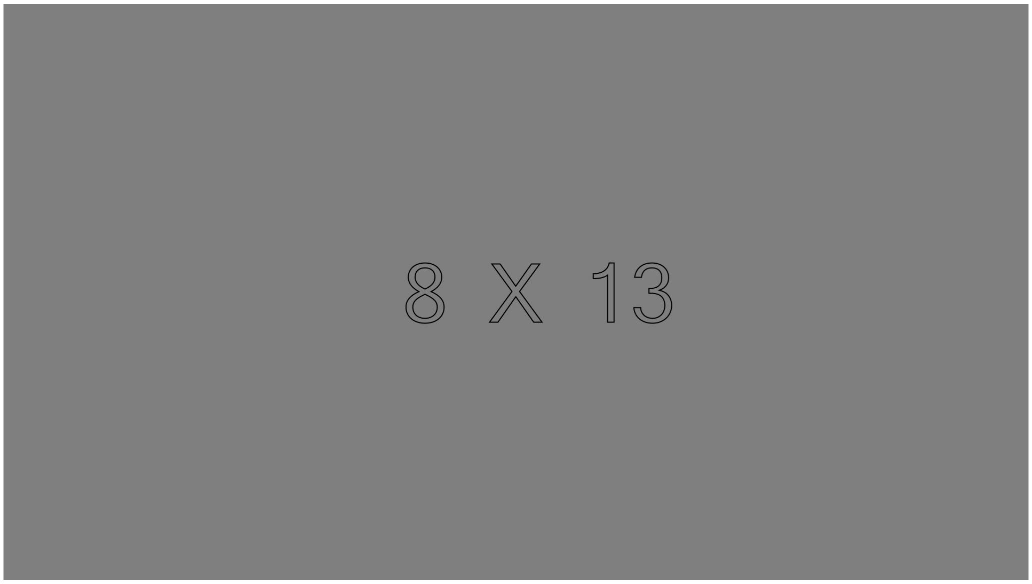


*Figure F.3: Multiplication slide of 8x13 during Task 2.*

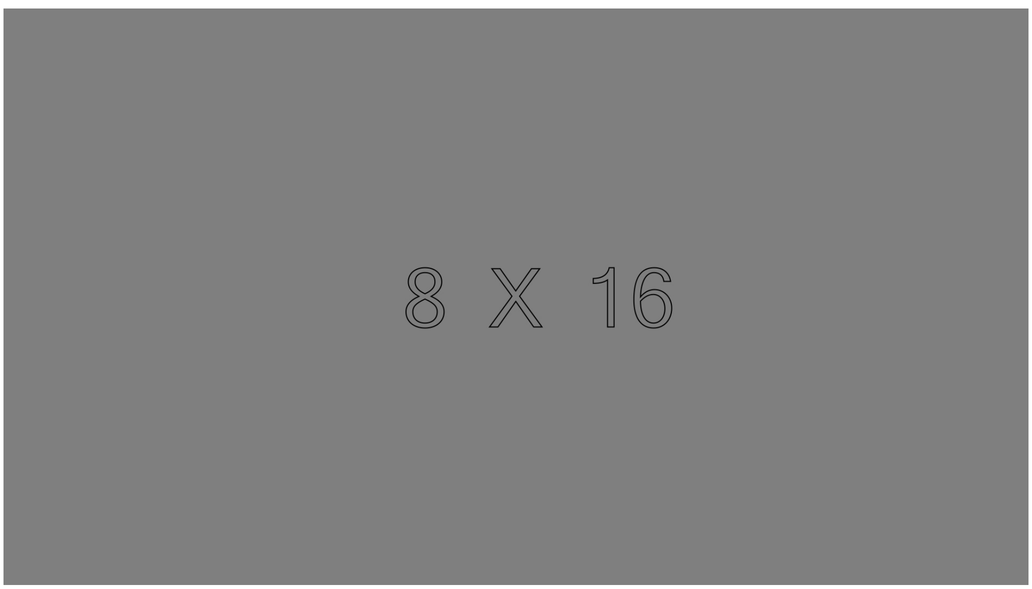


*Figure F.4: Multiplication slide of 8x16 during Task 2.*

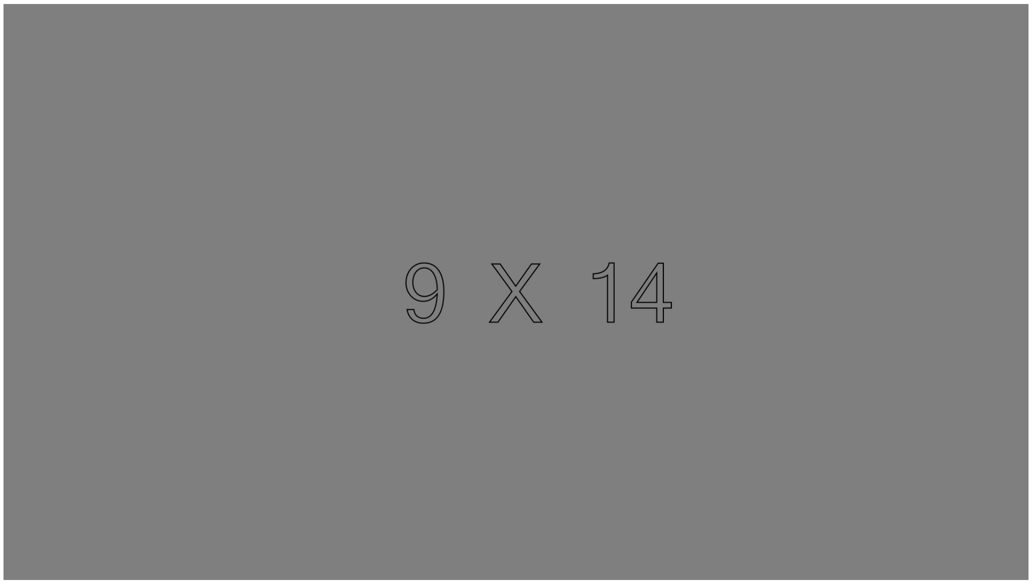


*Figure F.5: Multiplication slide of 9x14 during Task 2.*

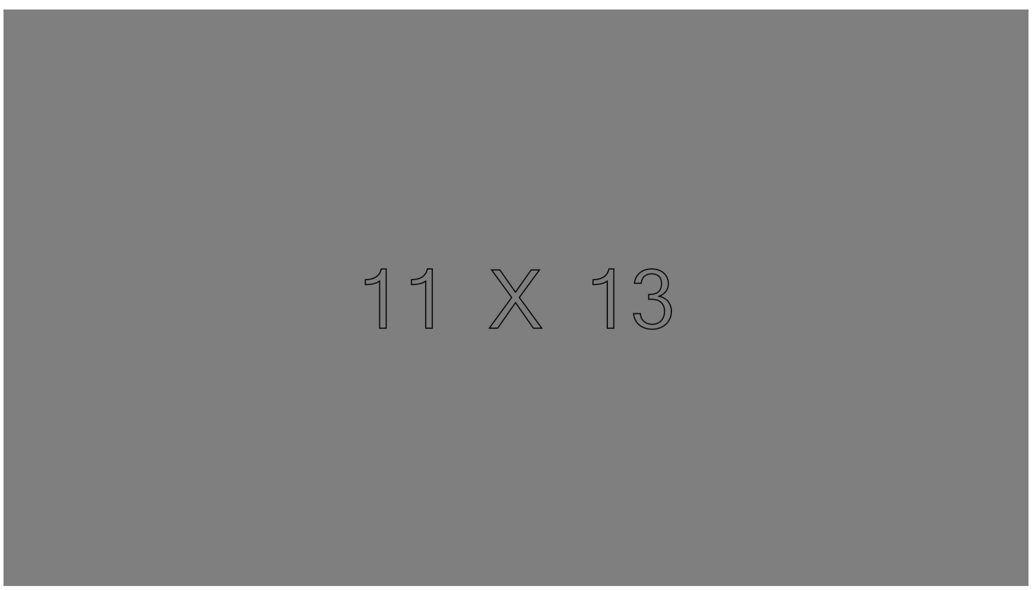


*Figure F.6: Multiplication slide of 11x13 during Task 2.*

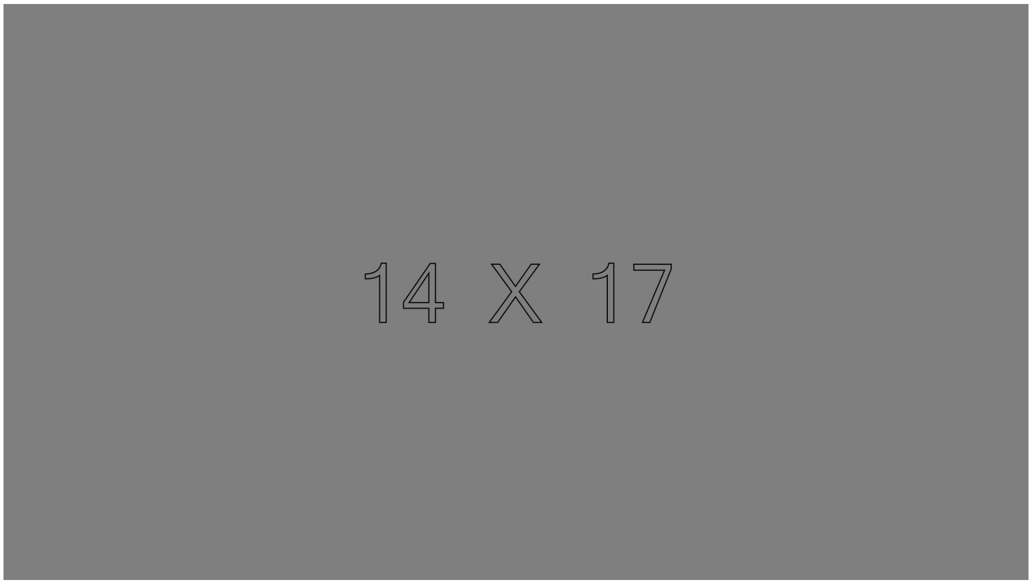


*Figure F.7: Multiplication slide of 14x17 during Task 2.*

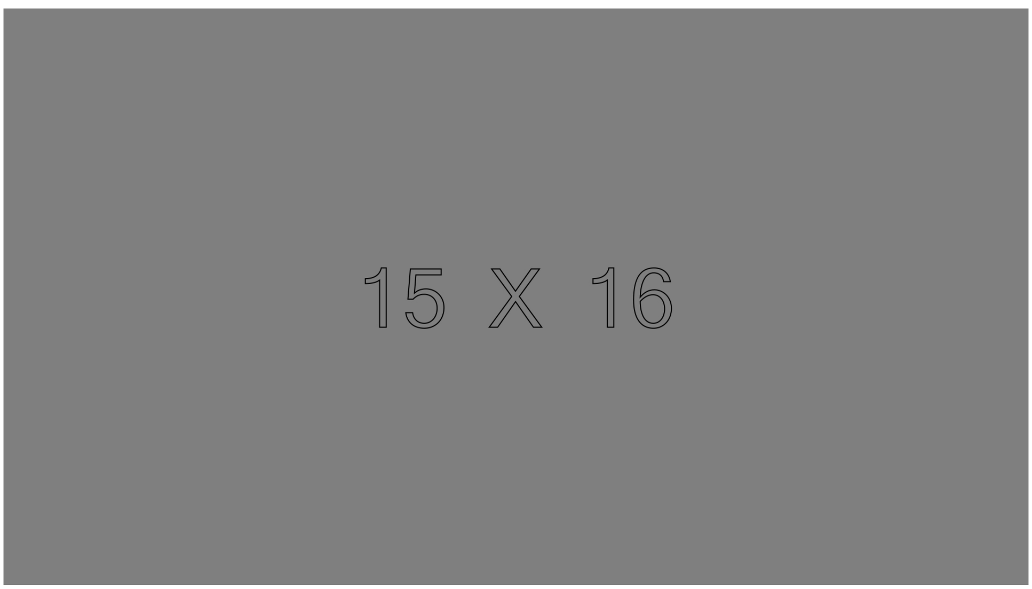


*Figure F.8: Multiplication slide of 15x16 during Task 2.*

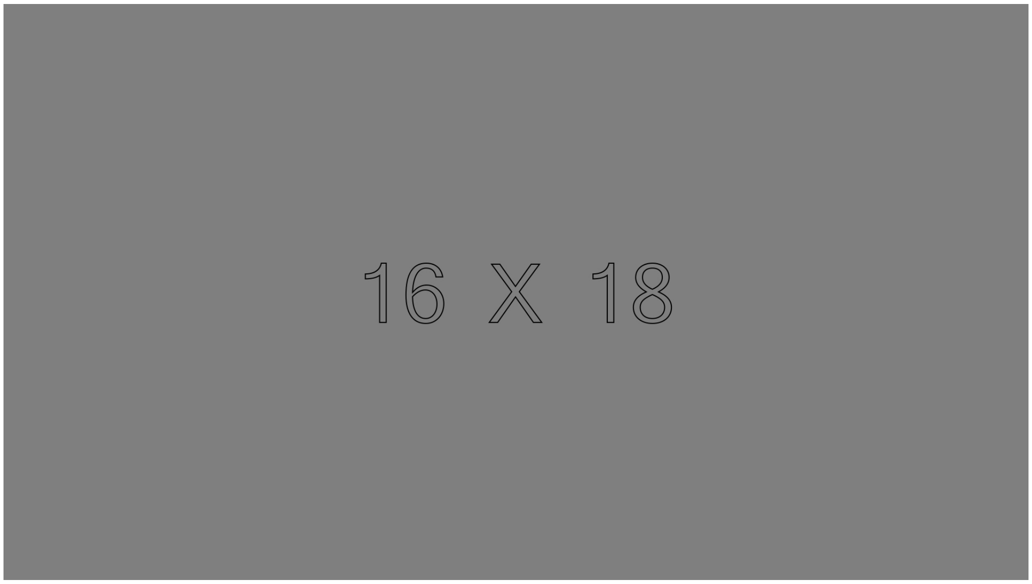


*Figure F.9: Multiplication slide of 16x18 during Task 2.*

# G Appendix G

*Figure G.1: Background of level 1, no road, during Task 3.*

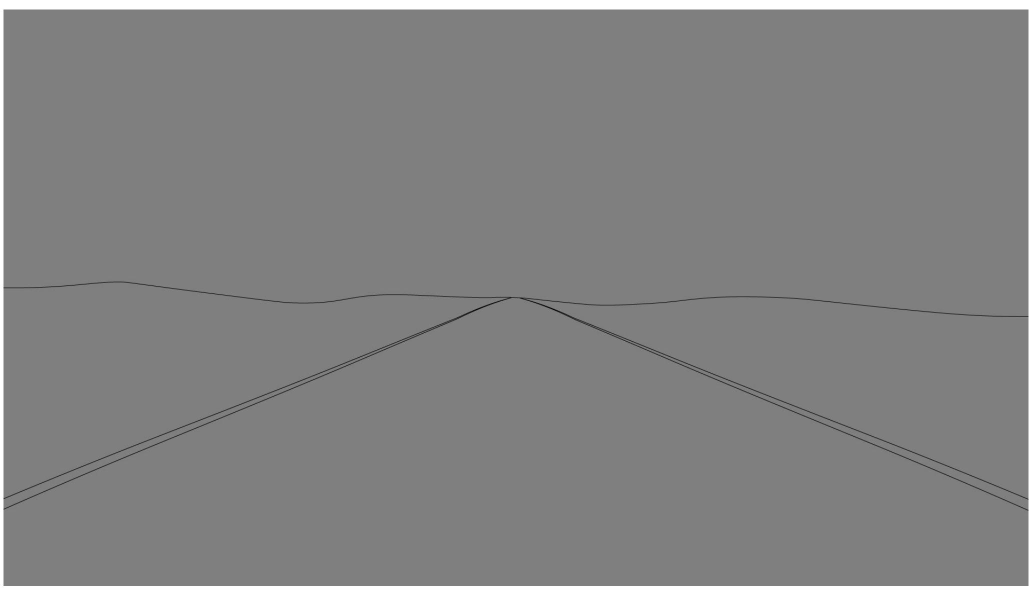

*Figure G.2: Background of level 2, simple road, during Task 3.*

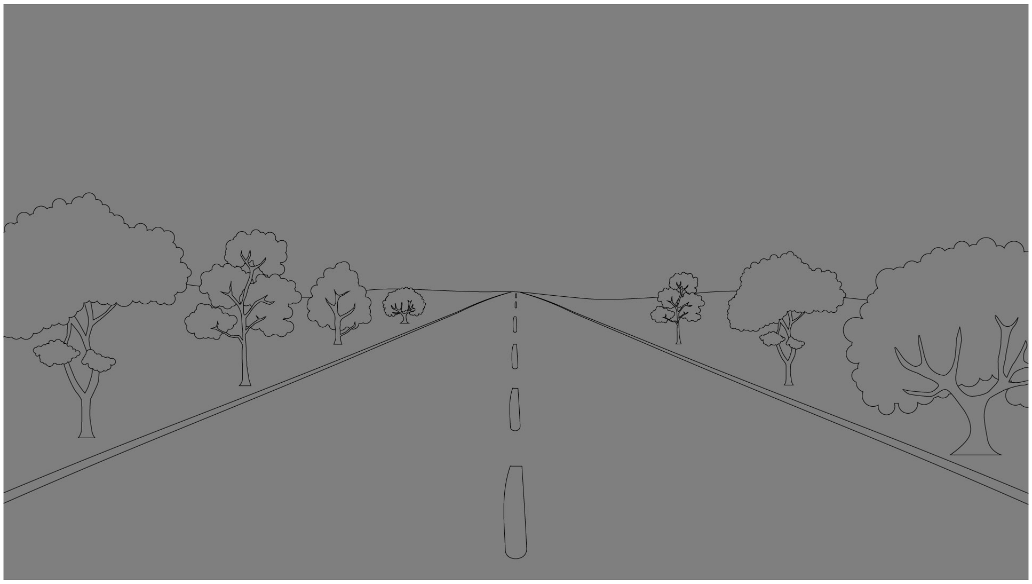

*Figure G.3: Background of level 3, average road, during Task 3.*

*Figure G.4: Background of level 4, realistic road, during Task 3.*

# H Appendix H

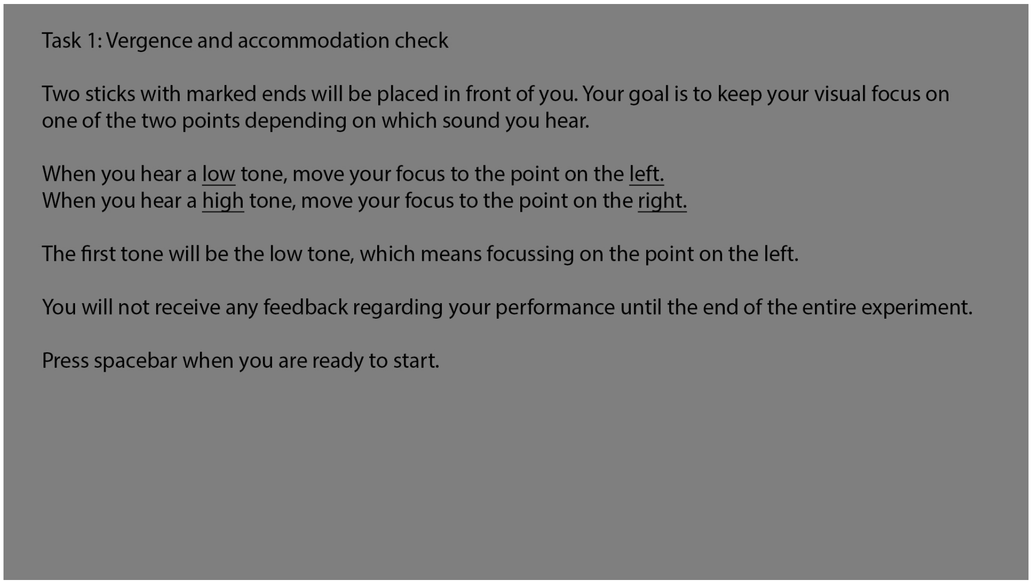


*Figure H.1: Instruction slide task 1*

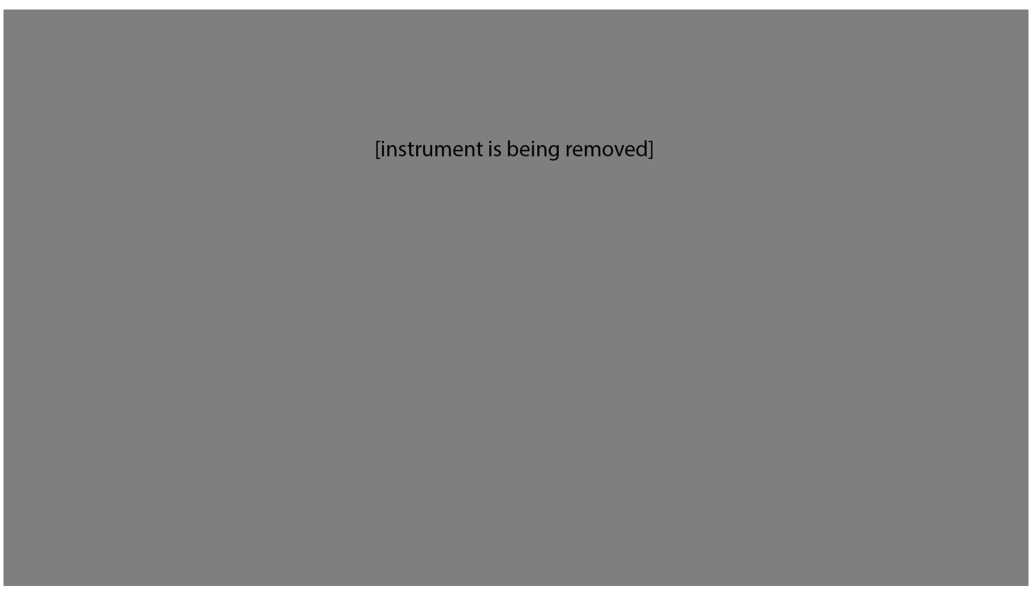


*Figure H.2: Instruction slide containing the message that the object of task 1 is being removed*

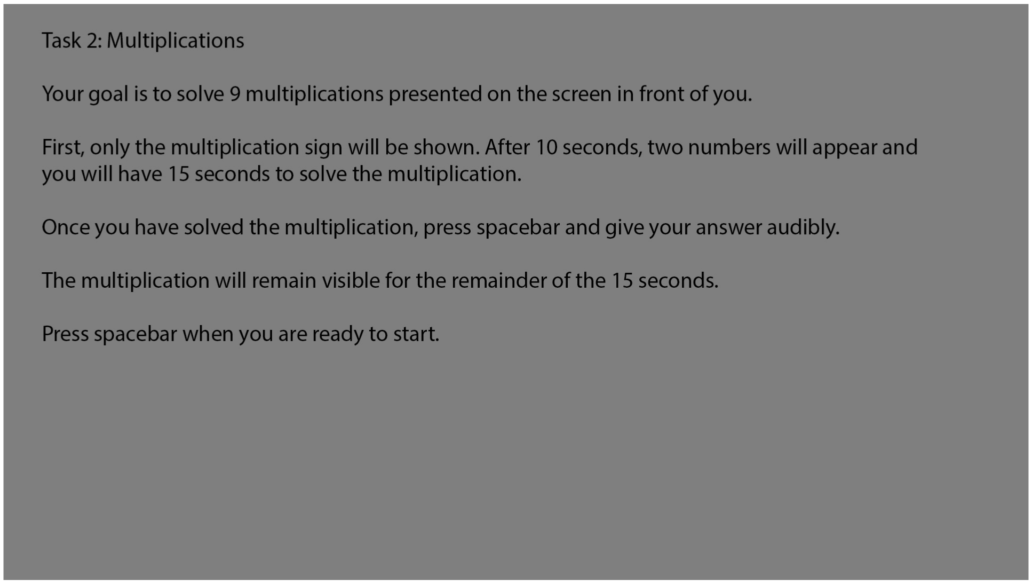


*Figure H.3: Instruction slide task 2*

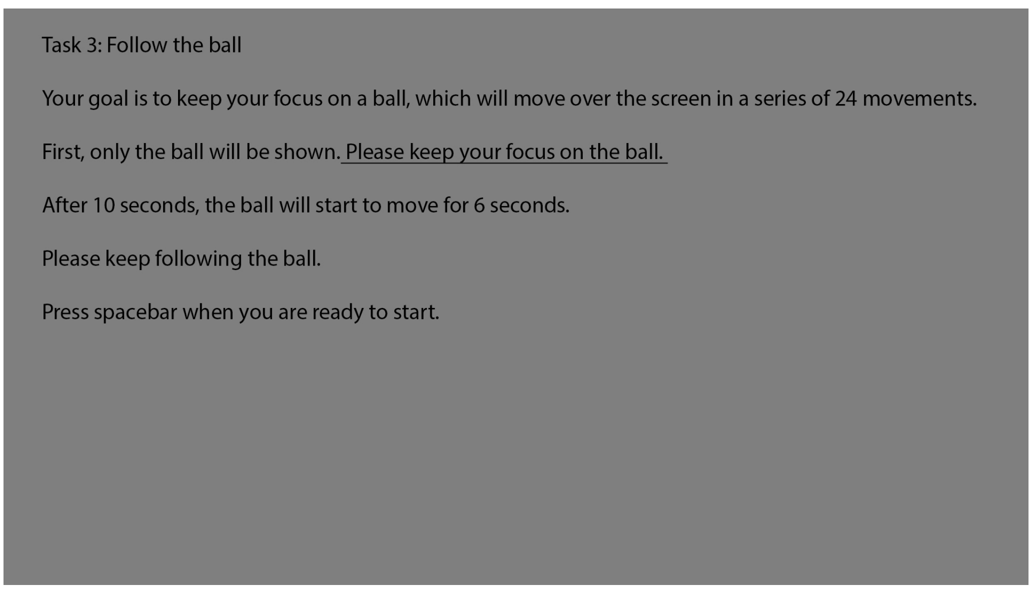


*Figure H.4: Instruction slide task 3*

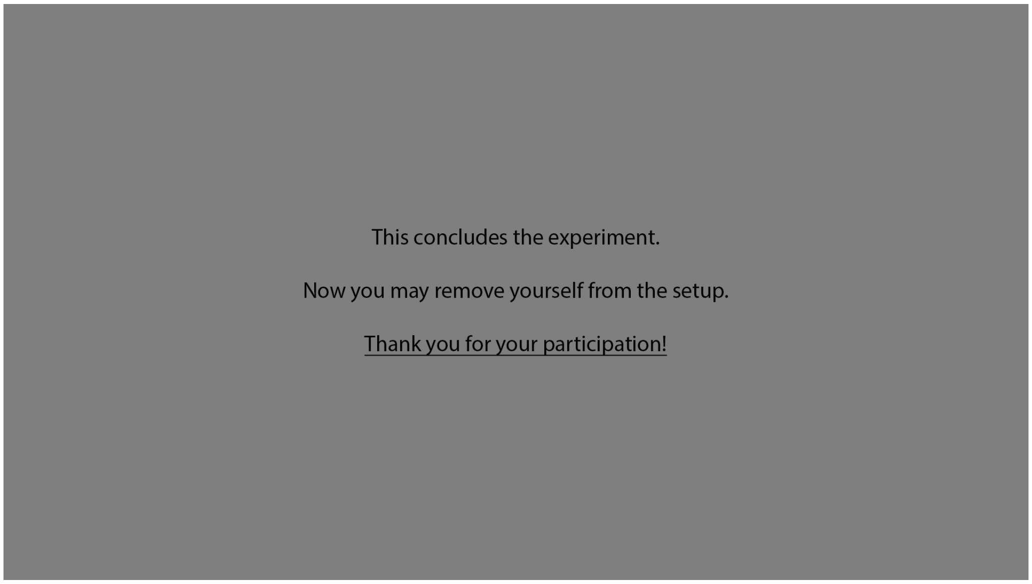


*Figure H.5: Instruction slide containing the message of thanking the participant for their participation*

# I Appendix I

*Table 11: List of all the variables coming out of the PoweRef III. These variables are presented as columns in the CVS-file.*

| | |
|---|---|
| 1 | Time [ms] |
| 2 | Segment |
| 3 | Distance() [m] |
| 4 | Brightness() [percent] |
| 5 | PupilFound(L) |
| 6 | PupilPos(L).X() |
| 7 | PupilPos(L).Y() |
| 8 | PupilSizeMM(L).X() [mm] |
| 9 | PupilSizeMM(L).Y() [mm] |
| 10 | PupilDiameterMM(L) [mm] |
| 11 | PupilBrightness(L) [DU] |
| 12 | PurkinjeSize(L).X() [px] |
| 13 | PurkinjeSize(L).Y() [px] |
| 14 | Decentration(L).X() [px] |
| 15 | Decentration(L).Y() [px] |
| 16 | Gaze(L).X() [degree] |
| 17 | Gaze(L).Y() [degree] |
| 18 | Refraction(L).Sphere() [Dpt] |
| 19 | GetAverageRefraction(L).Contents() |
| 20 | GetAverageRefraction(L).Mean() [Dpt] |
| 21 | GetAverageRefraction(L).StdDeviation() [Dpt] |
| 22 | PupilFound(R) |
| 23 | PupilPos(R).X() |
| 24 | PupilPos(R).Y() |
| 25 | PupilSizeMM(R).X() [mm] |
| 26 | PupilSizeMM(R).Y() [mm] |
| 27 | PupilDiameterMM(R) [mm] |
| 28 | PupilBrightness(R) [DU] |
| 29 | PurkinjeSize(R).X() [px] |
| 30 | PurkinjeSize(R).Y() [px] |
| 31 | Decentration(R).X() [px] |
| 32 | Decentration(R).Y() [px] |
| 33 | Gaze(R).X() [degree] |
| 34 | Gaze(R).Y() [degree] |
| 35 | Refraction(R).Sphere() [Dpt] |
| 36 | GetAverageRefraction(R).Contents() |
| 37 | GetAverageRefraction(R).Mean() [Dpt] |
| 38 | GetAverageRefraction(R).StdDeviation() [Dpt] |
| 39 | InterpupDist() [mm] |
| 40 | InterpupAngle() [degree] |
| 41 | Trigger |

*Table 12: List of all the variables selected by the MATLAB-script readRaw. This is processed by the next MATLAB-script, processRawData.*

| | |
|---|---|
| 1 | Time [ms] |
| 2 | PupilFound(L) |
| 3 | PupilSizeMM(L).X() [mm] |
| 4 | PupilSizeMM(L).Y() [mm] |
| 5 | PupilDiameterMM(L) [mm] |
| 6 | Gaze(L).X() [degree] |
| 7 | Gaze(L).Y() [degree] |
| 8 | Refraction(L).Sphere() [Dpt] |
| 9 | PupilFound(R) |
| 10 | PupilSizeMM(R).X() [mm] |
| 11 | PupilSizeMM(R).Y() [mm] |
| 12 | PupilDiameterMM(R) [mm] |
| 13 | Gaze(R).X() [degree] |
| 14 | Gaze(R).Y() [degree] |
| 15 | Refraction(R).Sphere() [Dpt] |
| 16 | Trigger |

*Table 13: List of all the variables used by the MATLAB-script processRawData. Analysis is done by the following MATLAB-script, StatisticsFiltData.*

| | |
|---|---|
| 1 | PupilDiameterMM(L) [mm] |
| 2 | Gaze(L).X() [degree] |
| 3 | Gaze(L).Y() [degree] |
| 4 | Refraction(L).Sphere() [Dpt] |
| 5 | PupilDiameterMM(R) [mm] |
| 6 | Gaze(R).X() [degree] |
| 7 | Gaze(R).Y() [degree] |
| 8 | Refraction(R).Sphere() [Dpt] |

# J Appendix J

## J.1 Cross-correlation right and left eye task 1

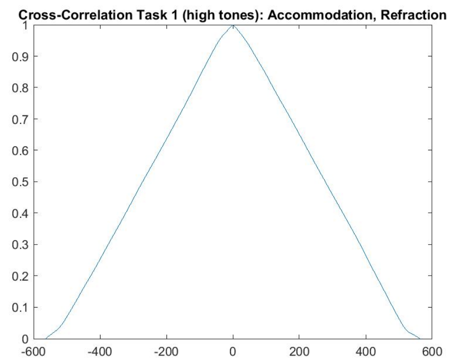


*Figure J.1: Cross-correlation task 1 high tones: accommodation*

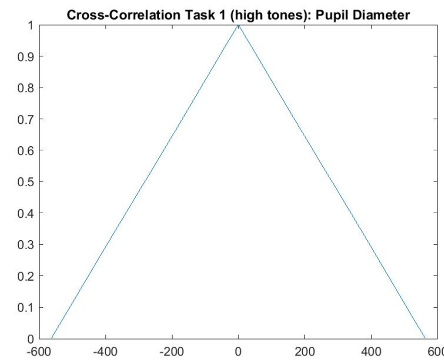


*Figure J.2: Cross-correlation task 1 high tones: Pupil diameter*

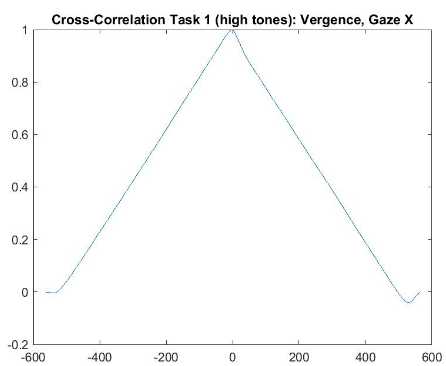


*Figure J.3: Cross-correlation task 1 high tones: Vergence, Gaze X*

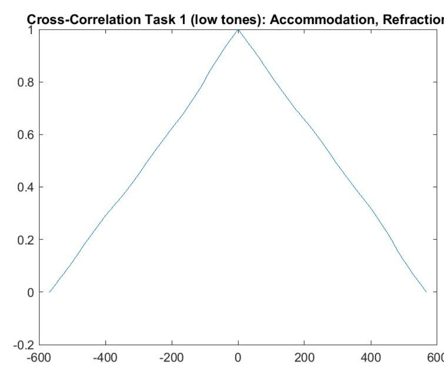


*Figure J.4: Cross-correlation task 1 low tones: accommodation*

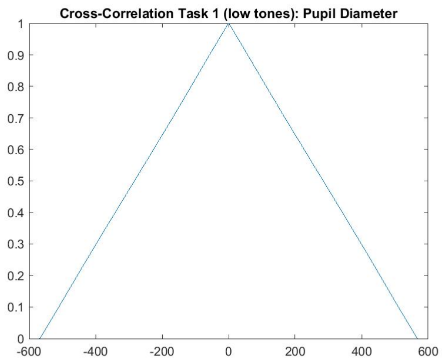


*Figure J.5: Cross-correlation task 1 low tones: Pupil diameter*

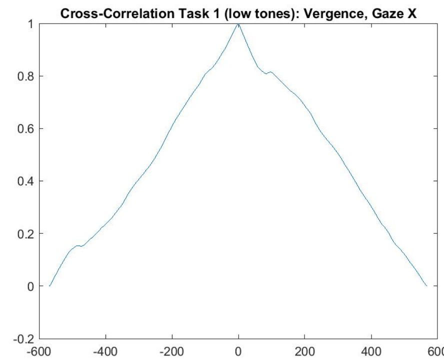


*Figure J.6: Cross-correlation task 1 low tones: Vergence, gaze X*

## J.2 Cross-correlation right and left eye task 2

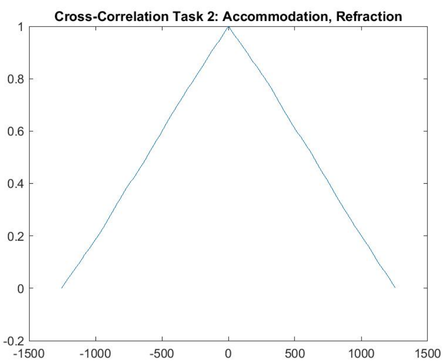


*Figure J.7: Cross-correlation task 2: accommodation*

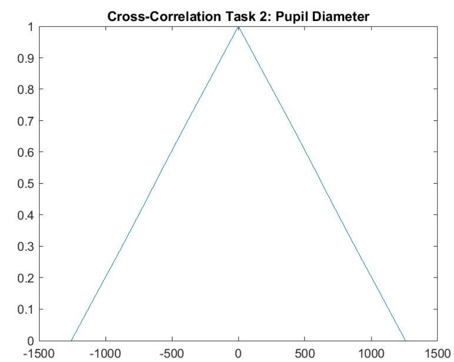


*Figure J.8: Cross-correlation task 2: Pupil diameter*

## J.3 Cross-correlation right and left eye task 3

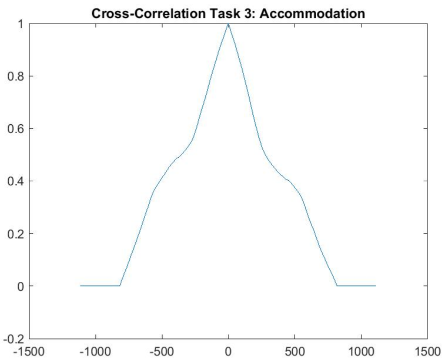


*Figure J.9: Cross-correlation task 3: accommodation*

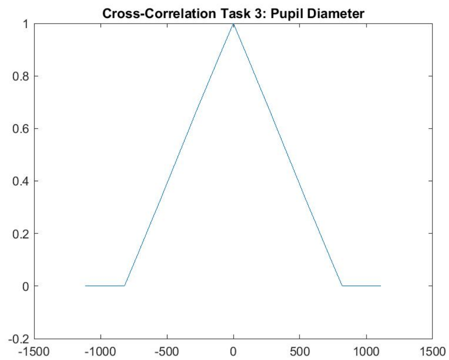


*Figure J.10: Cross-correlation task 3: Pupil diameter*

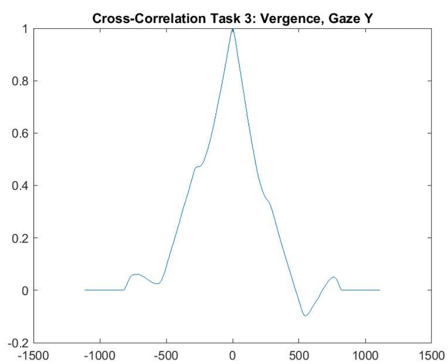


*Figure J.11: Cross-correlation task 3: Vergence, Gaze Y*

# K Appendix K

## K.1 Scatter plots Task 2

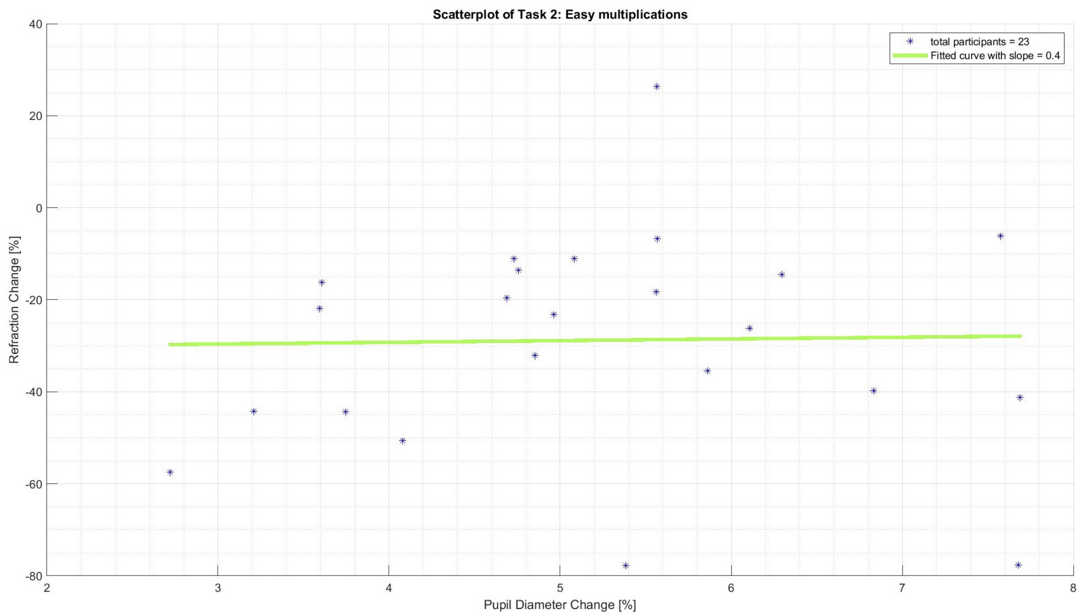


*Figure K.1: Scatter plot task 2, easy multiplications*

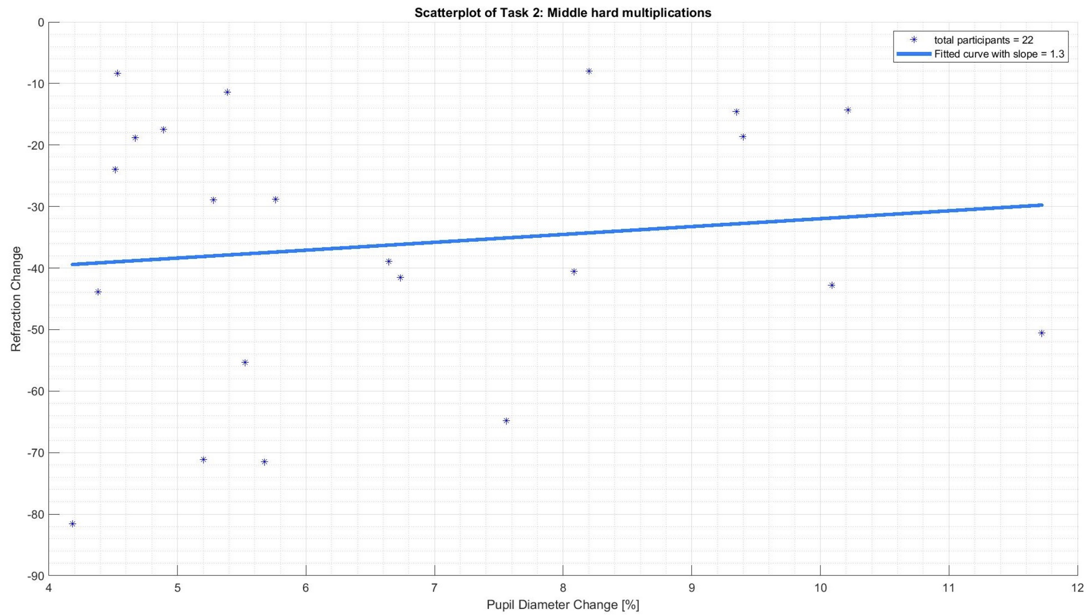


*Figure K.2: Scatter plot task 2, average multiplications*

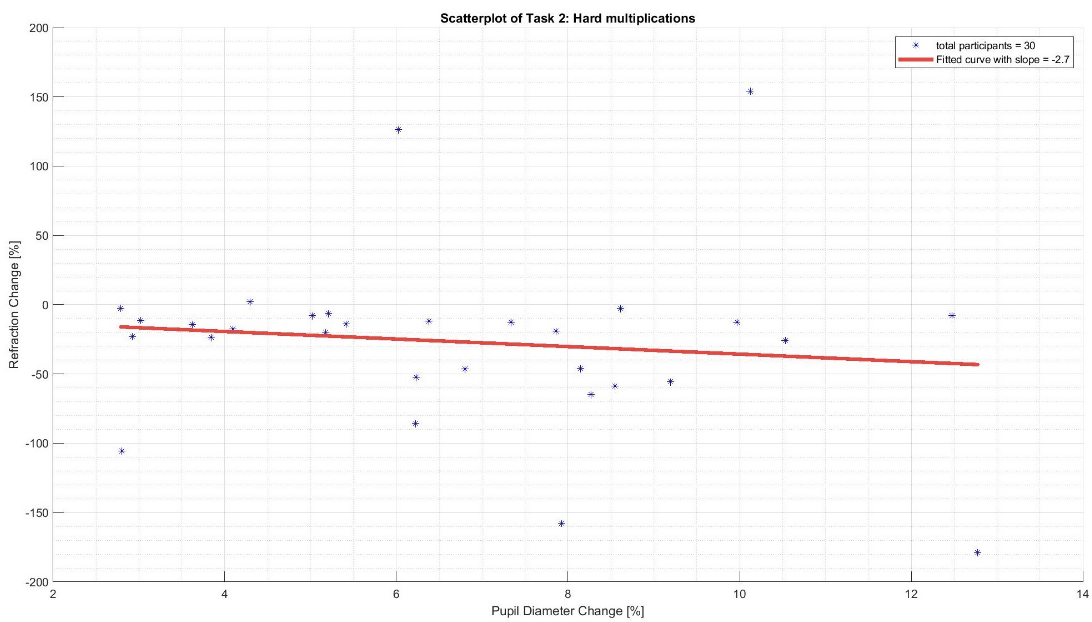


*Figure K.3: Scatter plot task 2, difficult multiplications*

## K.2 Scatter plots Task 3

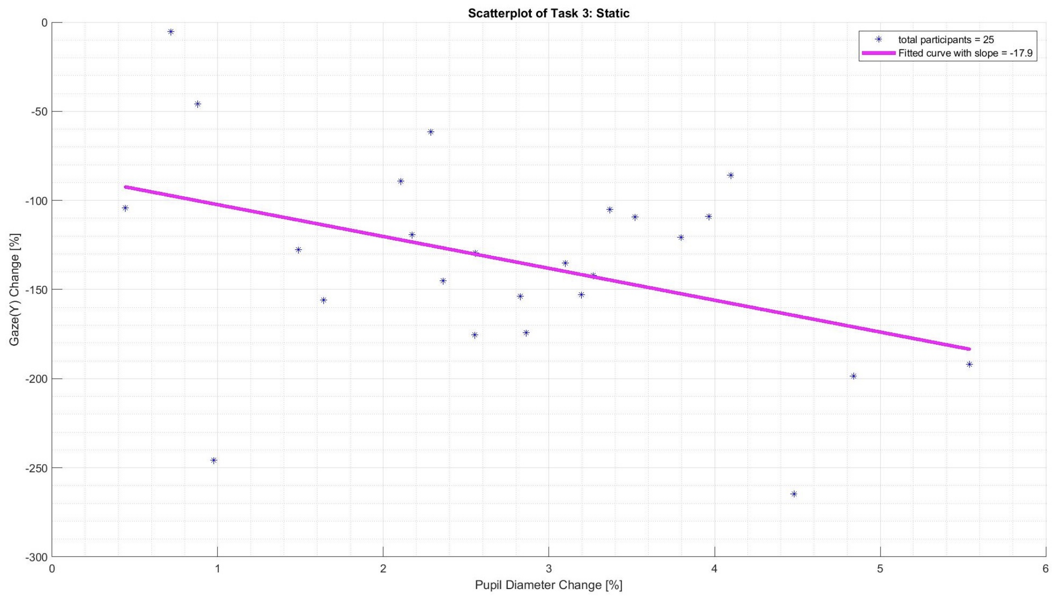


*Figure K.4: Scatter plot task 3, static*

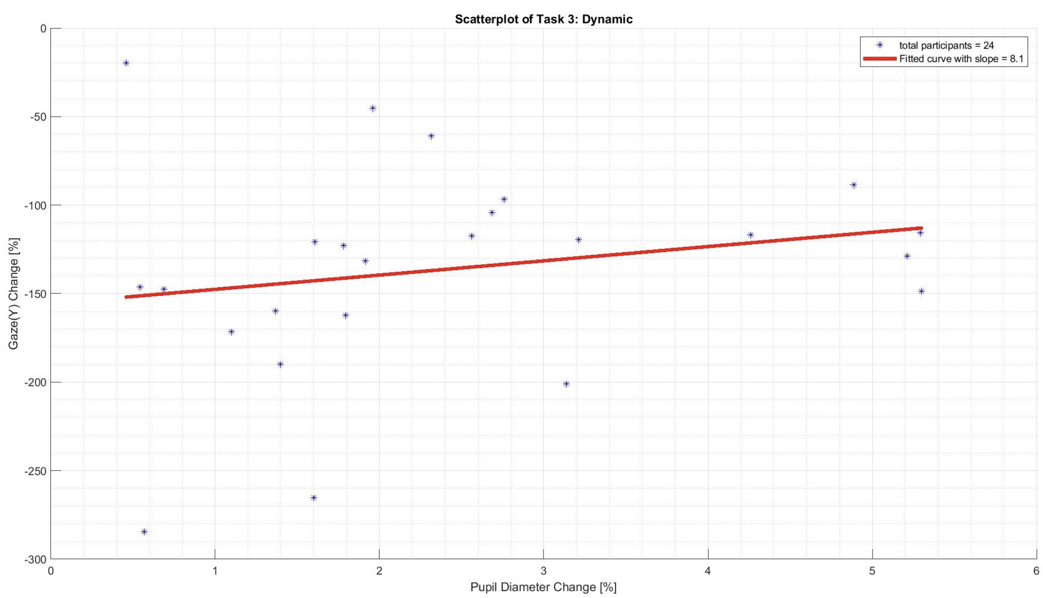


*Figure K.5: Scatter plot task 3, dynamic*

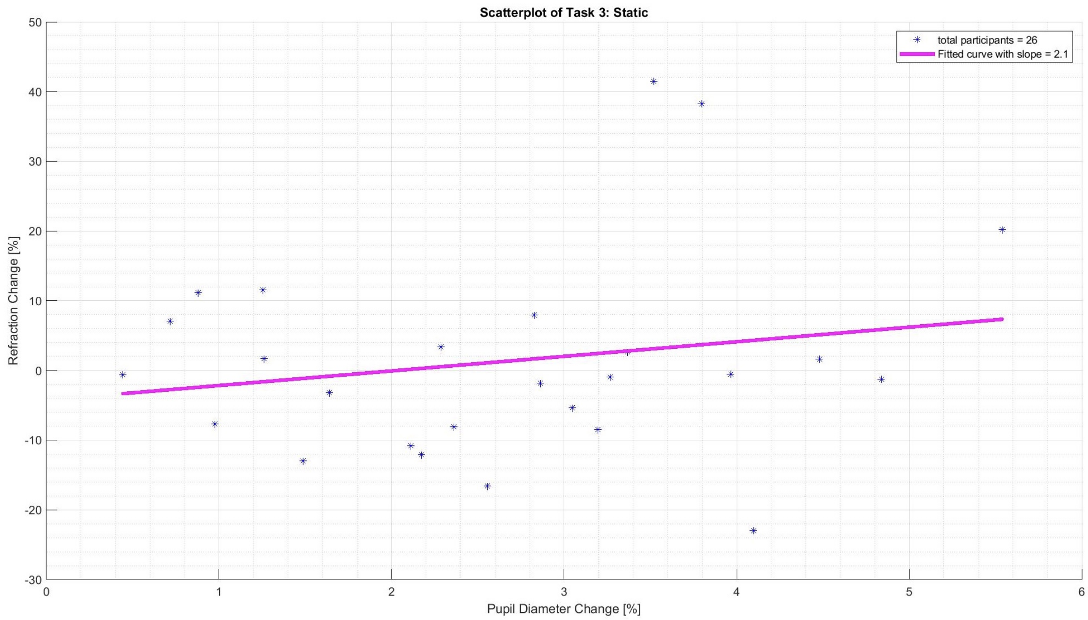


*Figure K.6: Scatter plot task 3, static*

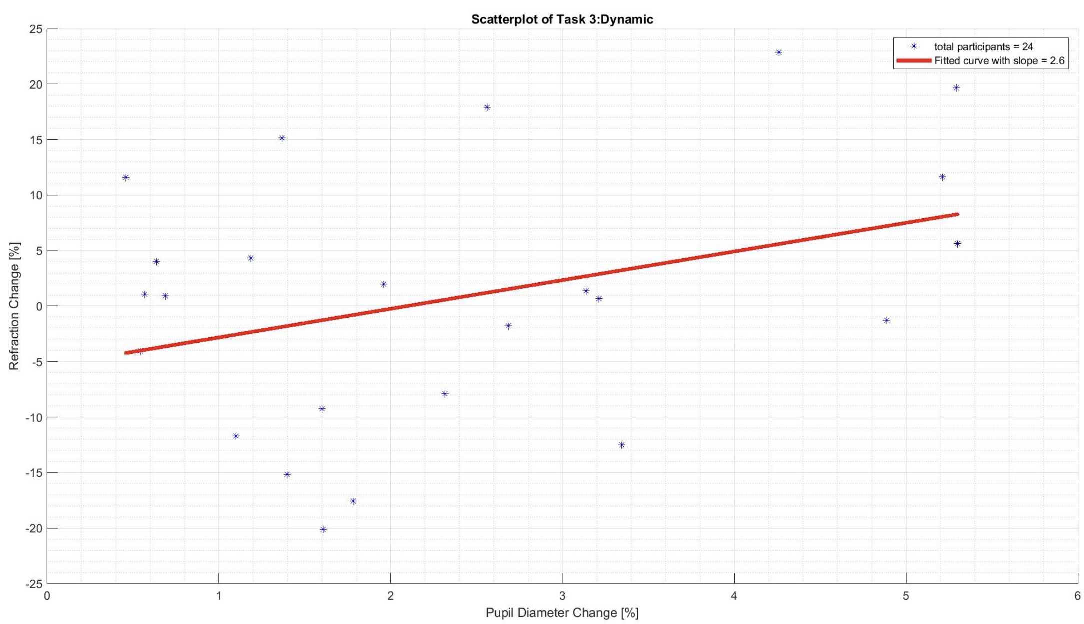


*Figure K.7: Scatter plot task 3, dynamic*

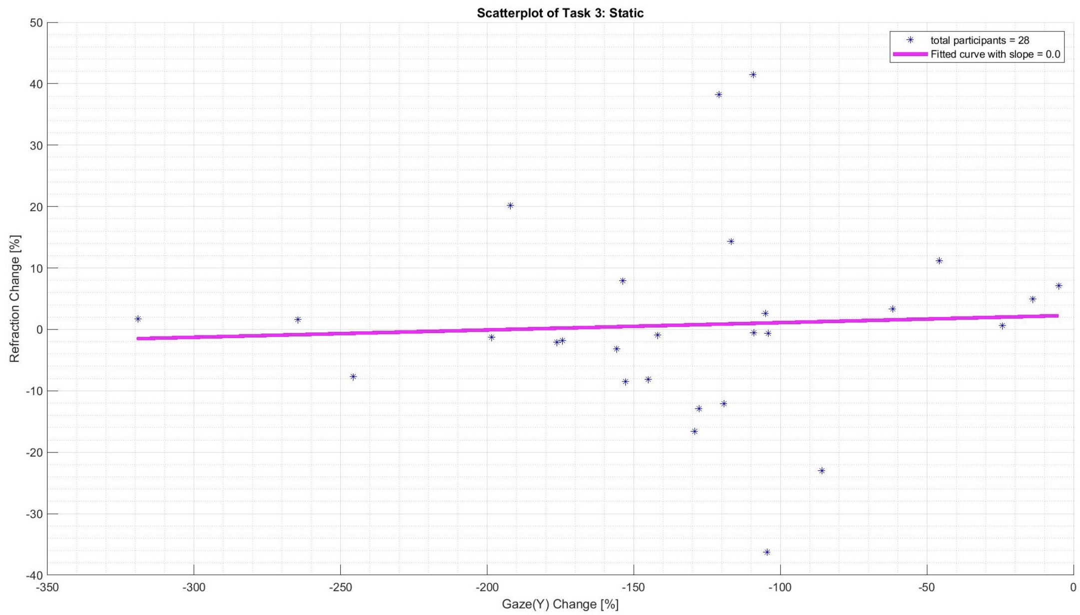


*Figure K.8: Scatter plot task 3, static*

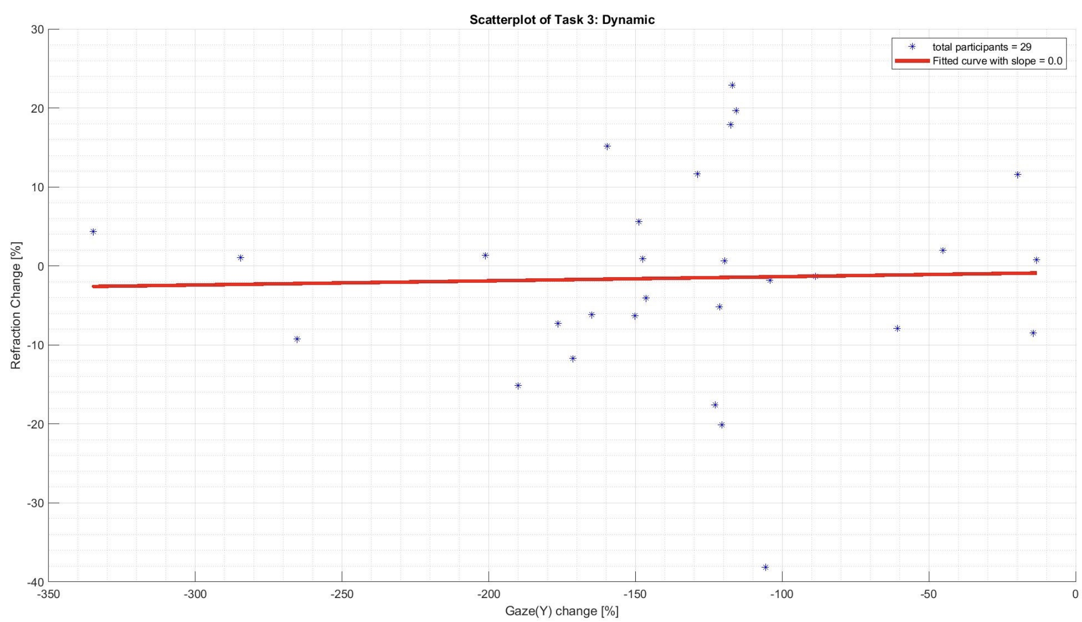


*Figure K.9: Scatter plot task 3, dynamic*

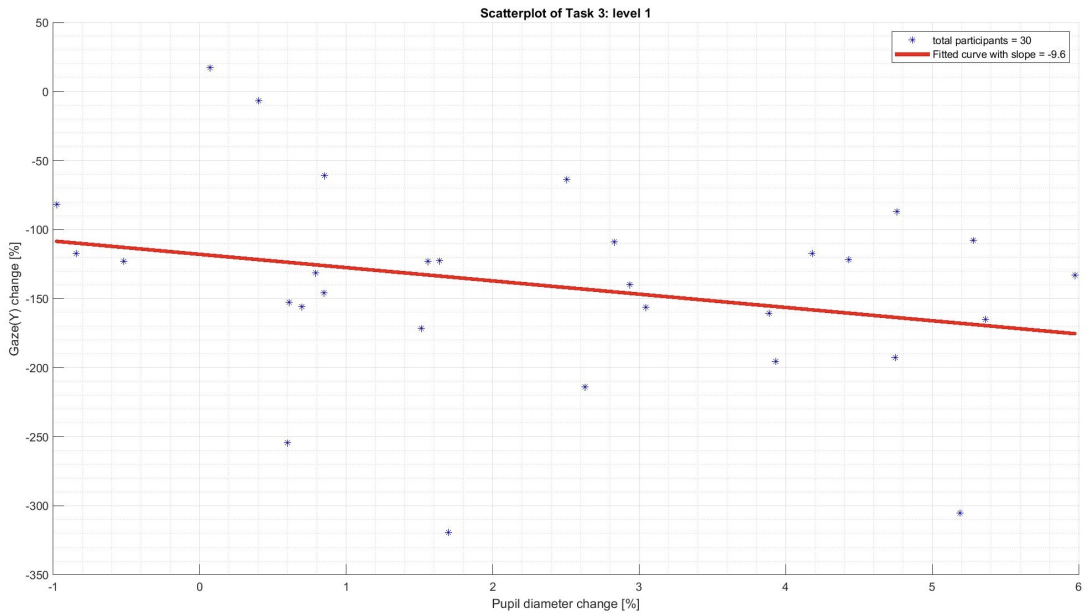


*Figure K.10: Scatter plot task 3, level 1*

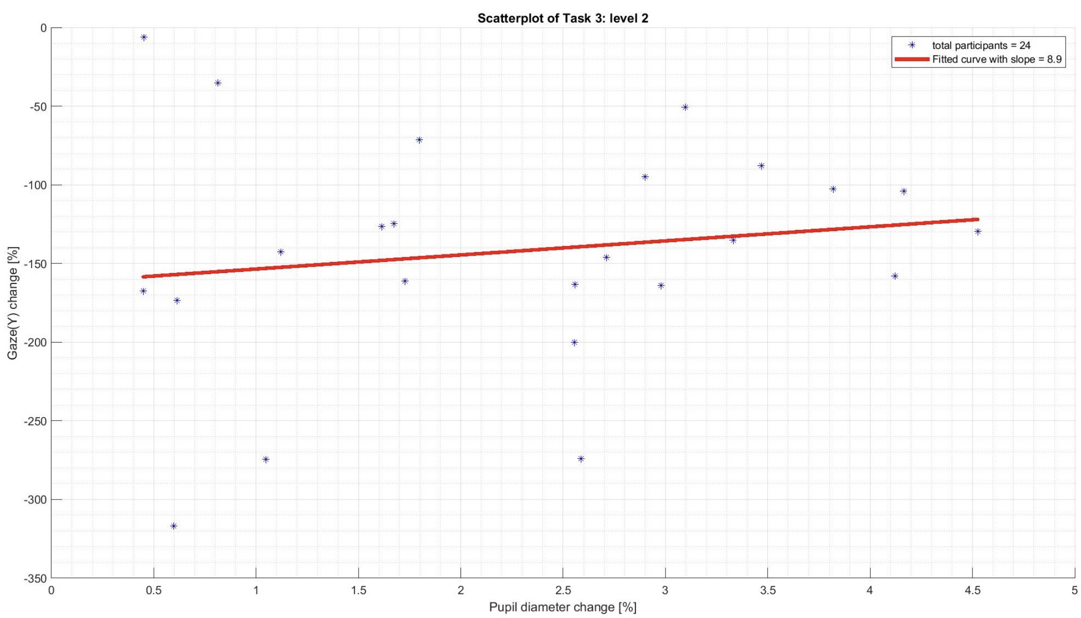


*Figure K.11: Scatter plot task 3, level 2*

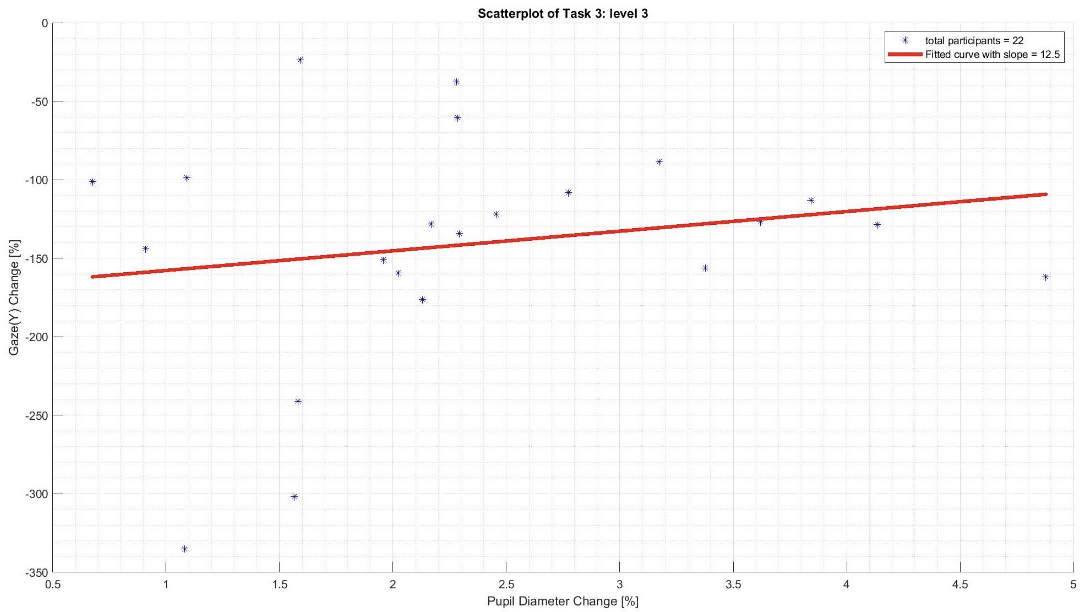


*Figure K.12: Scatter plot task 3, level 3*

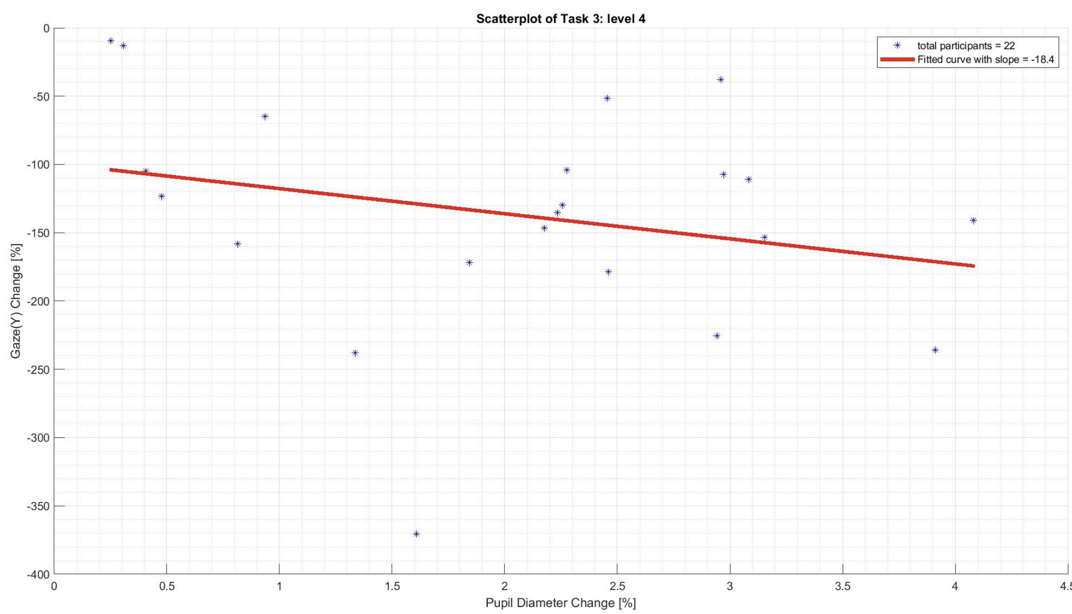


*Figure K.13: Scatter plot task 3, level 4*

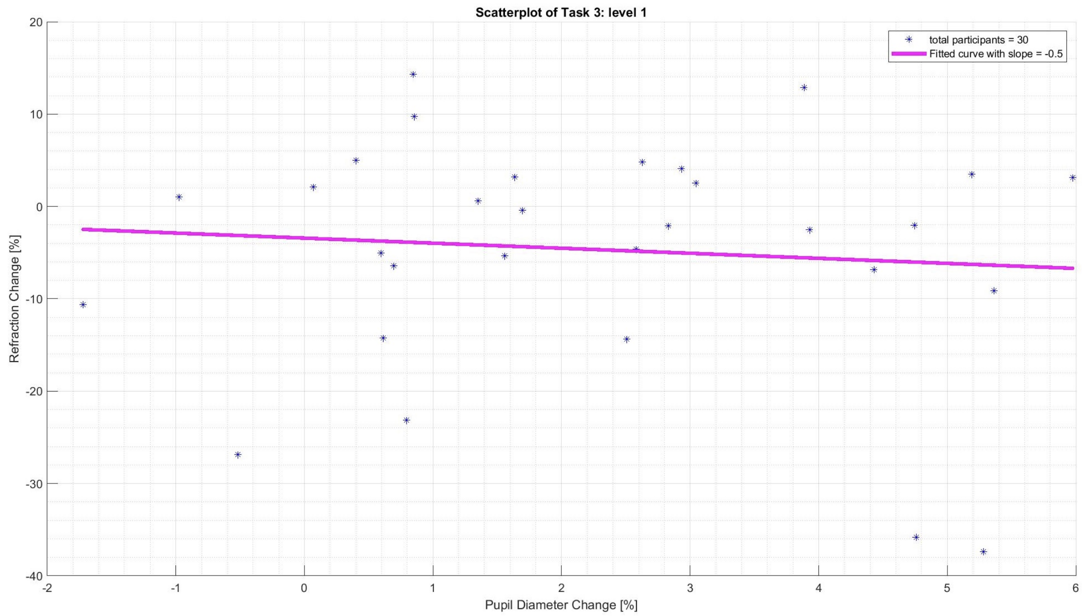


*Figure K.14: Scatter plot task 3, level 1*

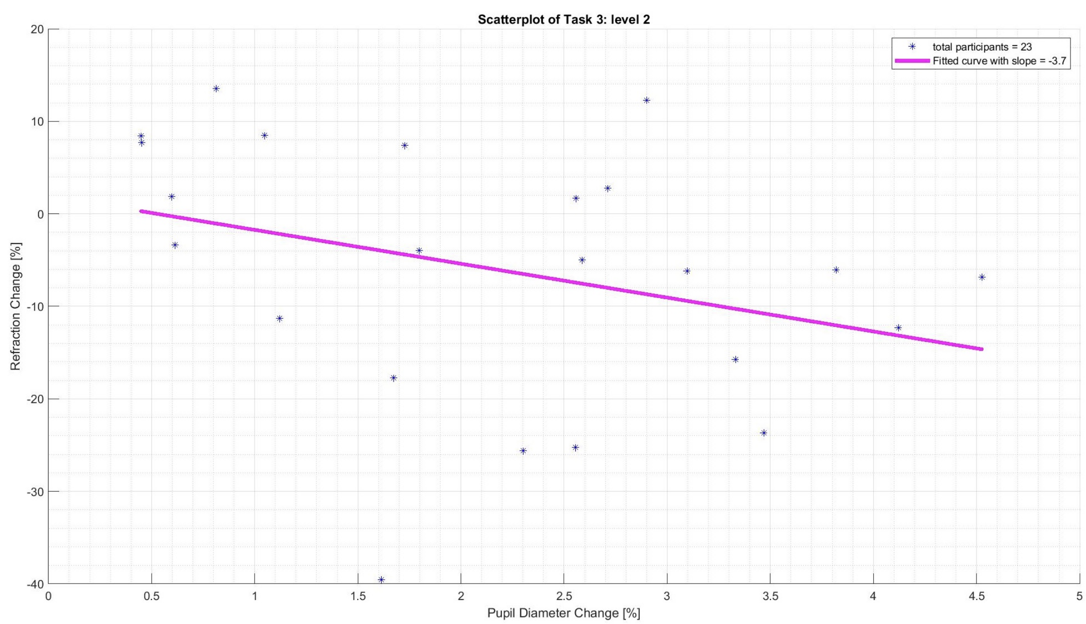


*Figure K.15: Scatter plot task 3, level 2*

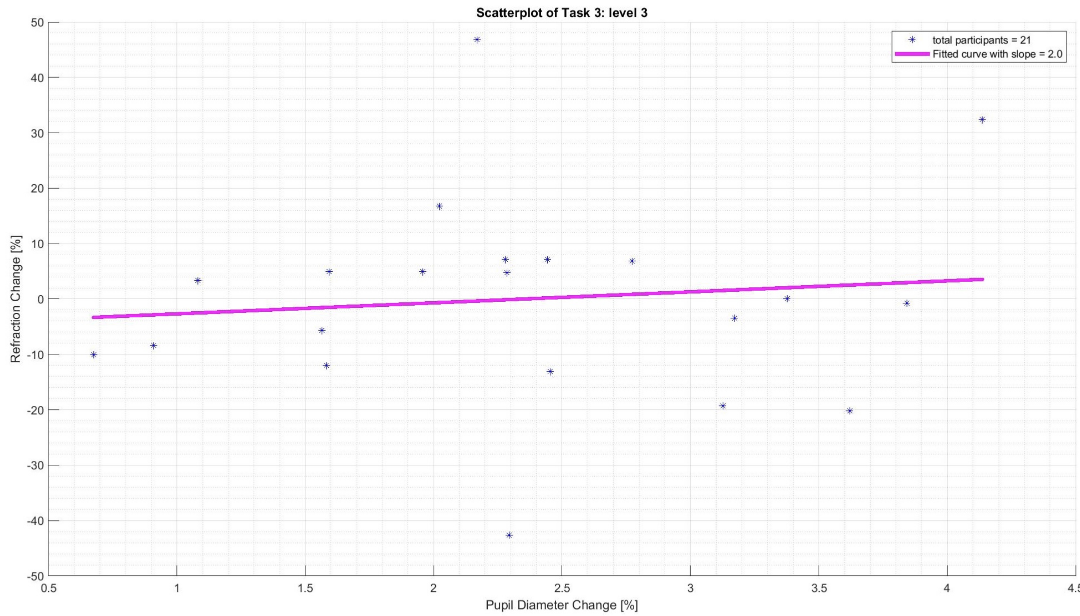


*Figure K.16: Scatter plot task 3, level 3*

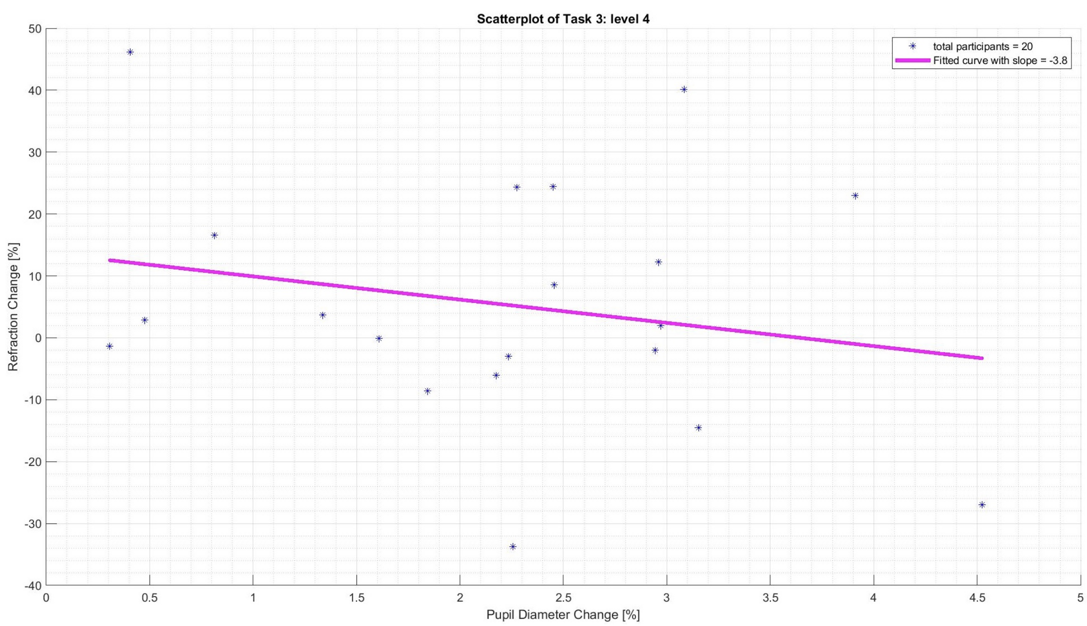


*Figure K.17: Scatter plot task 3, level 4*

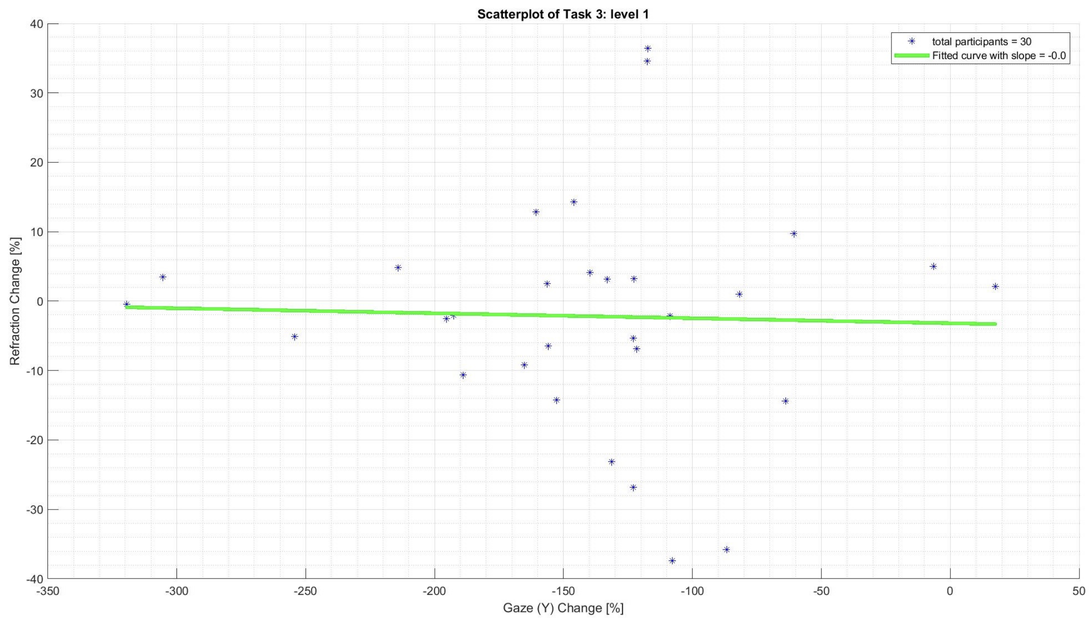


*Figure K.18: Scatter plot task 3, level 1*

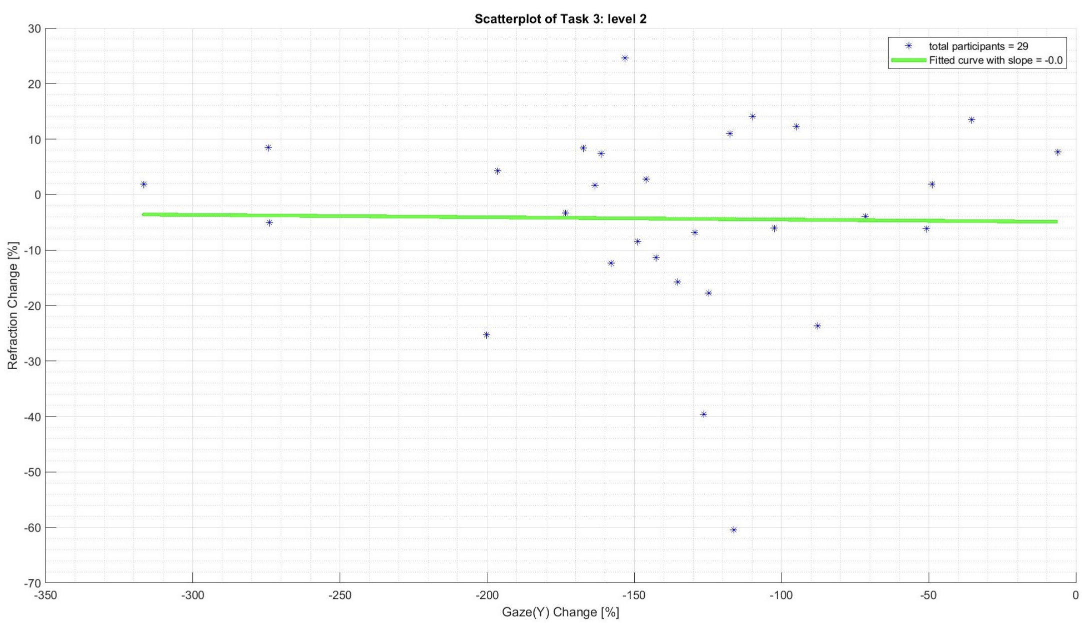


*Figure K.19: Scatter plot task 3, level 2*

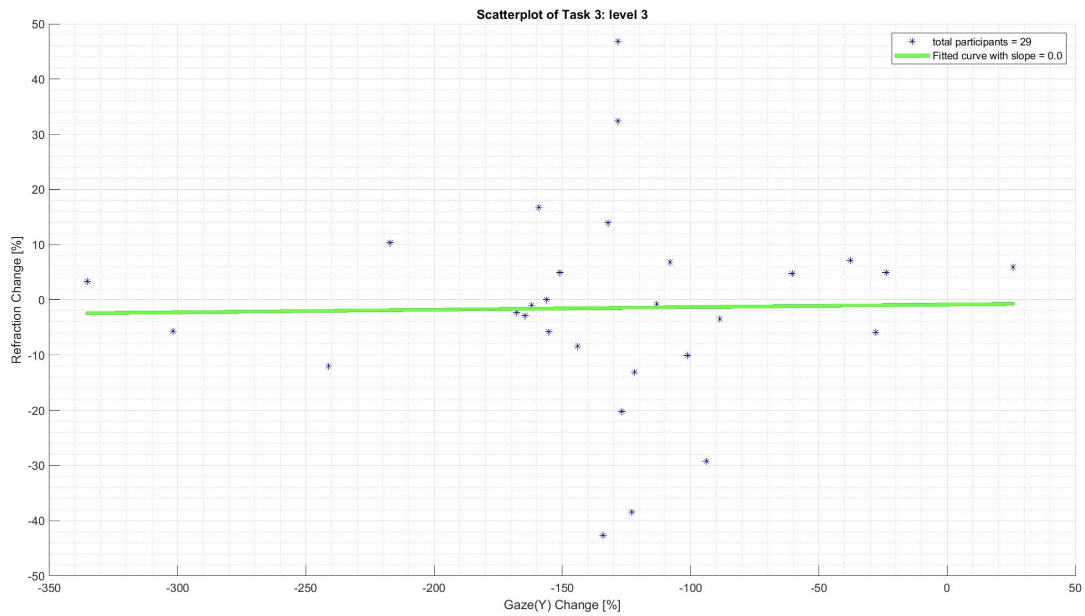


*Figure K.20: Scatter plot task 3, level 3*

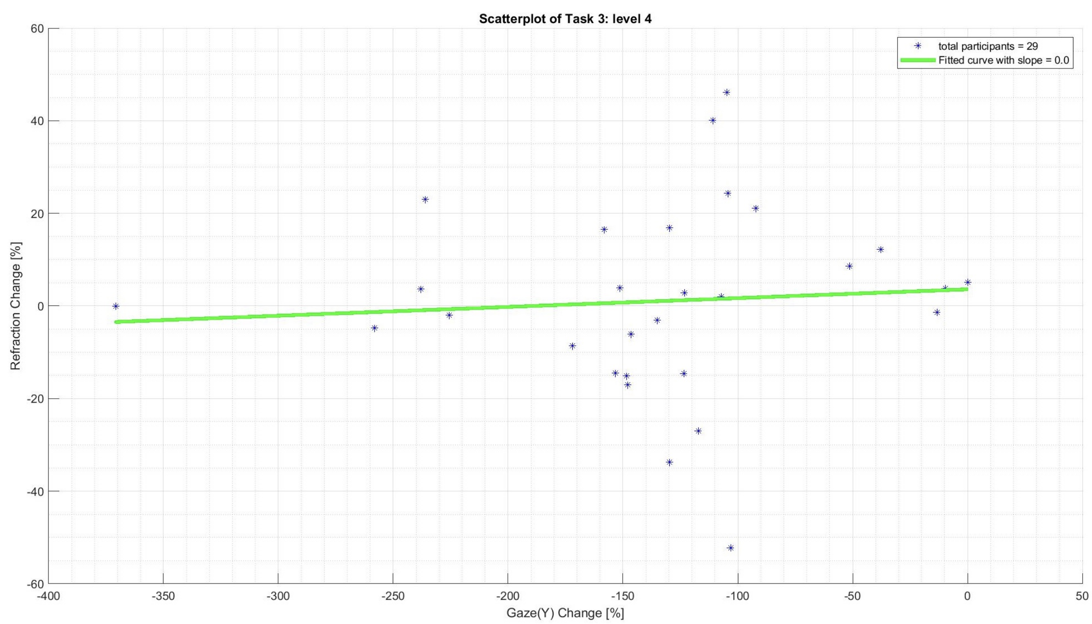


*Figure K.21: Scatter plot task 3, level 4*

# L Appendix L

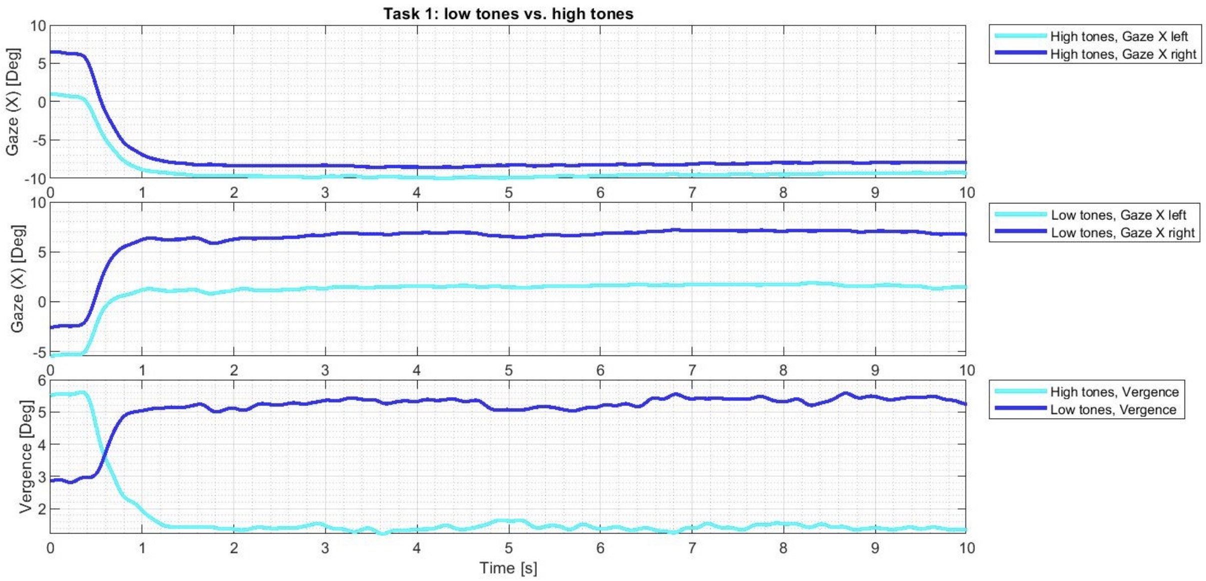


*Figure L.1: Subplot from task 1. Gaze(X) plotted for the high tones left and right, and Gaze(X) plotted for the low tones left and right. The vergence is calculated using the formula:*

$$Vergence = |(Gaze(X)righteye) - (Gaze(X)lefteye)|$$

## References


[1] D.J. Benjamin. "Redefine Statistical Significance". In: *Nature Human Behaviour* 2 (2018), pp. 6–10.

[2] Medicus Medical Company. *plusoptiX R09 PowerRef 3*. 2019. URL: `https://www.medicus.ua/eng/product/ophthalmic-equipment/plusoptiX-R09-PowerRef-3/manufacturer:8/print:1`.

[3] J.T. Enright. "Perspective vergence: oculomotor responses to line drawings". In: *Pergamon Journals Ltd* 27.9 (Sept. 1987), pp. 1513–1626.

[4] M. Feil, B. Moser, and M Abegg. "The interaction of pupil response with the vergence system". In: *Graefe's Archive for Clinical and Experimental Ophthalmology* 255.11 (Aug. 2017), pp. 2247–2253.

[5] M.T. Swantston W.C. Gogel. "Perceived size and motion in depth from optical expansion". In: *Perception : Psychophysics* 39.5 (Feb. 1986), pp. 309–326.

[6] E.B. Goldstein. *Sensation and perception*. Wadsworth-Thomson Learning, 2002.

[7] E.H. Hess. *The tell-tale eye: How your eyes reveal hidden thoughts and emotions*. Van Nostrand Reinhold, 1975.

[8] E.H. Hess and J.M. Polt. "Pupil Size as Related to Interest Value of Visual Stimuli". In: *Science* 132.3423 (Aug. 1960), pp. 349–350.

[9] E.H. Hess and J.M. Polt. "Pupil Size in Relation to Mental Activity during Simple Problem-Solving". In: *Science* 143.3611 (Mar. 1964), pp. 1190–1192.

[10] G.K. Hung, J.L. Semmlon, and K.J. Ciuffreda. "The Near Response: Modeling, Instrumentation, and Clinical Applications". In: *IEEE Transactions on Biomedical Engineering* 31.12 (Dec. 1984), 910–919.

[11] L. Kooijman J. De Winter S.M. Petermeijer and D. Dodou. "Replicating the studies of Eckhard Hess". in press 2019.

[12] S. Jainta, J. Hoormann, and W. Jaschinski. "Ocular accommodation and cognitive demand: An additional indicator besides pupil size and cardiovascular measures?" In: *Journal of Negative Results in BioMedicine* 7.1 (2008).

[13] M.A. Just and P.A. Carpenter. "The intensity dimension of thought: Pupillometric indices of sentence processing". In: *Canadian Journal of Experimental Psychology* 47.2 (June 1993), pp. 310–339.

[14] D. Kahneman and J. Beatty. "Pupil diameter and load on memory". In: *Science* 154.3756 (Dec. 1966), pp. 1583–1585.

[15] S. Mathot. "Pupillometry: Psychology, Physiology, and Function". In: *Journal of Cognition* 16 (Feb. 2018), pp. 1–23.

[16] U. Sulutvedt, T.K. Mannix, and B. Laeng. "Gaze and the Eye Pupil Adjust to Imagined Size and Distance". In: *Cognitive Science* 42.8 (Nov. 2018), 3159–3176.

[17] G. Audesirk T. Audesirk and B.E. Byers. *Biologiy, life on earth with physiology*. Pearson, 2014.

[18] G. Westheimer. "Accommodation Measurements in Empty Visual Fields". In: *Journal of the optical society of America* 47.8 (1957), pp. 714–718.