



Methodology for Assessing Cognitive Workload through Pupillary Oculometrics Under Degraded Visual Environment

Leonard Temme, Christopher Aura, Kevin O'Brien, Aaron McAtee, Cristina Delgado-Howard, Hillary Plummer, & Paul St. Onge

DISTRIBUTION STATEMENT A. Approved for public release; distribution unlimited.

Notice

Qualified Requesters

Qualified requesters may obtain copies from the Defense Technical Information Center (DTIC), Fort Belvoir, Virginia 22060. Orders will be expedited if placed through the librarian or other person designated to request documents from DTIC.

Change of Address

Organizations receiving reports from the U.S. Army Aeromedical Research Laboratory on automatic mailing lists should confirm correct address when corresponding about laboratory reports.

Disposition

Destroy this document when it is no longer needed. Do not return it to the originator.

Disclaimer

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other official documentation. Citation of trade names in this report does not constitute an official Department of the Army endorsement or approval of the use of such commercial items.

Human Subjects Use Statement

In the conduct of research involving human subjects, the investigator(s) adhered to the policies regarding the protection of human subjects as prescribed by Department of Defense Instruction 3216.02 (Protection of Human Subjects and Adherence to Ethical Standards in DoD-Supported Research) dated 8 November 2011.

IRB Determination and Number

This data set was collected under the U.S. Army Medical Research and Materiel Command, now the U.S. Army Medical Research and Development Command, Institutional Review Board approved protocol #10751 and U.S. Army Aeromedical Research Laboratory (USAARL) protocol #2018-010.

REPORT DOCUMENTATION PAGE					Form Approved OMB No. 0704-0188							
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.												
1. REPORT DATE (DD-MM-YYYY) 01-09-2021		2. REPORT TYPE Technical Report			3. DATES COVERED (From - To)							
4. TITLE AND SUBTITLE Methodology for Assessing Cognitive Workload through Pupillary Oculometrics under Degraded Visual Environment				5a. CONTRACT NUMBER								
				5b. GRANT NUMBER								
				5c. PROGRAM ELEMENT NUMBER								
6. AUTHOR(S) Temme, L. ¹ , Aura, C. ¹ , O'Brien, K. ¹ , McAtee, A. ^{1,2} , Delgado-Howard, C. ^{1,3} , Plummer, H. ^{1,3} , & St. Onge, P. ¹				5d. PROJECT NUMBER 21000								
				5e. TASK NUMBER								
				5f. WORK UNIT NUMBER								
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Aeromedical Research Laboratory P.O. Box 620577 Fort Rucker, AL 36362					8. PERFORMING ORGANIZATION REPORT NUMBER USAARL-TECH-TR--2021-38							
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Medical Research and Development Command 504 Scot Street Fort Detrick, MD 21702-5202					10. SPONSOR/MONITOR'S ACRONYM(S) USAMRDC							
					11. SPONSOR/MONITOR'S REPORT NUMBER(S)							
12. DISTRIBUTION/AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A. Approved for public release; distribution unlimited.												
13. SUPPLEMENTARY NOTES ¹ U.S. Army Aeromedical Research Laboratory, ² Goldbelt Frontier, LLC., ³ Oak Ridge Institute for Science and Education												
14. ABSTRACT This report provides a detailed description of the instrumentation supporting USAARL's study titled 'Assessing Cognitive Workload through Pupillary Oculometrics under Degraded Visual Environments.' The appendices include the software code used to collect the data. The report also presents sample raw data traces of a single volunteer's complete dataset to illustrate the quality of the data. One of the purposes of the present report is to encourage and facilitate further use of the data since the dataset may be available to qualified members of the wider research community. The full data set demonstrates that commercial off the shelf (COTS) eye-tracking technology has potential as a non-invasive tool to collect objective, quantitative indices of arousal, fatigue, mental workload, and other neuropsychological states of Army aviators.												
15. SUBJECT TERMS pupil, oculometrics, degraded visual environments, eye-tracking, mental workload, fatigue												
16. SECURITY CLASSIFICATION OF: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 33%; padding: 2px;">a. REPORT</td> <td style="width: 33%; padding: 2px;">b. ABSTRACT</td> <td style="width: 33%; padding: 2px;">c. THIS PAGE</td> </tr> <tr> <td style="text-align: center; padding: 2px;">UNCLAS</td> <td style="text-align: center; padding: 2px;">UNCLAS</td> <td style="text-align: center; padding: 2px;">UNCLAS</td> </tr> </table>			a. REPORT	b. ABSTRACT	c. THIS PAGE	UNCLAS	UNCLAS	UNCLAS	17. LIMITATION OF ABSTRACT SAR		18. NUMBER OF PAGES 89	
a. REPORT	b. ABSTRACT	c. THIS PAGE										
UNCLAS	UNCLAS	UNCLAS										
			19a. NAME OF RESPONSIBLE PERSON Loraine St. Onge, PhD									
			19b. TELEPHONE NUMBER (Include area code) 334-255-6906									

This page is intentionally blank.

Summary

The present report documents the methods, instrumentation and procedures USAARL's Warfighter Protection Group (WPG) used to assess the pupil's responses to well-defined, time-varying luminance conditions coincident with the performance of cognitively demanding tasks. Large dynamic changes in luminance are inherent to military operational environments and these luminance changes actively drive pupil size (PS). However, tonus of the autonomic nervous system (ANS) also affects PS. Thus, PS behavior reflects the influence of fatigue, arousal, anxiety, mental or cognitive workload (CW), and other factors that affect the balance of the sympathetic and parasympathetic nervous systems. Importantly, the literature has yet to describe adequately how the PS reflects the combined effects of dynamic luminance changes combined with the impact of cognitive task performance. The differentiation between the two types of pupillary responses—luminance vs. ANS driven—is crucial for using PS as a possible indication of CW. Differentiating these two types of PS responses may be particularly challenging since the PS response to luminance changes can be three orders of magnitude greater than the PS responses to changes in ANS tonus.

This report provides a detailed description of the instrumentation supporting USAARL's study titled 'Assessing Cognitive Workload through Pupillary Oculometrics under Degraded Visual Environments.' The appendices include the software code used to collect the data. The report also presents sample raw data traces of a single volunteer's complete dataset to illustrate the quality of the data. One of the purposes of the present report is to encourage and facilitate further use of the data since the dataset may be available to qualified members of the wider research community. The full data set demonstrates that commercial off the shelf (COTS) eye-tracking technology has potential as a non-invasive tool to collect objective, quantitative indices of arousal, fatigue, mental workload, and other neuropsychological states of Army aviators. Statistical analyses and discussion of the full data are available elsewhere (Aura et al., 2021).

This page is intentionally blank.

Acknowledgements

This research was supported in part by an appointment to the Postgraduate Research Program at the U.S. Army Aeromedical Research Laboratory administered by the Oak Ridge Institute for Science and Education through an interagency agreement between the U.S. Department of Energy and the U.S. Army Medical Research and Development Command.

This space is intentionally blank.

Table of Contents

	Page
Summary	iii
Introduction.....	1
Methods.....	2
Participants.....	2
Equipment	2
Software.	2
Visual display.....	2
Eye tracking.	2
Equipment Preparation and Calibration Procedure.....	3
Experimental Task Breakdown.....	4
Light stimulation procedure.	4
Cognitive task (n-back).....	4
Experimental Session Breakdown	5
Participant instruction and eye tracking calibration.....	5
Experimental session and data collection procedure.	6
Data storage.	7
Results.....	8
Discussion and Recommendations	13
References.....	15
Appendix A. Acronyms and Abbreviations.....	16
Appendix B. DVE-Pupil Study Participant Packet.....	17
Appendix C. Experimenter Checklist	28
Appendix D. NASA TLX	30
Appendix E. CODE: designLumSetsV2.m.....	31
Appendix F. CODE: LightLevelTest.m.....	36
Appendix G. CODE: LumDynPupilPractV1.m.....	42
Appendix H. CODE: LumDynPupilFullV3.m.....	54

List of Figures and Tables

Figure 1. SR Research EyeLink 1000, desktop system. The setup used in our configuration did not require the chin rest pictured here.....	3
Figure 2. Overall study design.	7
Figure 3. The pupil diameter in pixels as a function of time in seconds recorded at 500 Hz in the left eye of a single subject during the 0-back task.	10
Figure 4. As Figure 3 except these data were collected with the 1-back task.	11
Figure 5. The same as Figure 4 except these data were collected with the 2-back task.	12
Table 1. Levels of n-back difficulty.....	5

This page is intentionally blank.

Introduction

As military aircraft instrumentation continues to grow in complexity, challenging the capabilities of the individual aviator, cognitive workload (CW) becomes a commensurately critical dimension that can make the difference between success and failure (Hancock, 2006). Hence, a major concern for military aviation medical and human factors research is the management of CW in ways to enhance mission effectiveness and safety (Hancock, 2006). Thus, the U.S. Army Aeromedical Research Laboratory (USAARL) maintains an active CW research program. An important goal of this program is to develop methods to monitor CW in real time, under operational conditions without interfering in any way with operational task performance (Aura et al., 2021).

This ambitious goal has yet to be achieved because every success is partial at best, falling short of an ideal solution. Given the complexity and difficulty of monitoring CW, no single approach or technology is likely to suffice; rather, solutions are more likely to emerge from a combination of compatible synergistic technologies. Among the emerging, compatible technologies that have potential for noninvasive CW monitoring in real time are those that track the overt behavior of an operator's eyes. The scientific literature demonstrates a close relation between the behavior of an operator's eyes and cognitive task performance, which suggests that the eye's behavior can serve as an indicator of CW.

Modern video-based eye trackers can make precise, detailed recordings of the eye's behavior. The datasets these eye trackers provide are rich sources of multidimensional information. These eye trackers typically record, as a function of time, the eye's line of sight (LOS) which identifies the sequence of objects viewed in the visual field. They also record pupil size (PS) and can estimate eye blinks. The USAARL's current CW research program incorporates several different eye trackers for a variety of specific purposes.

Recently USAARL completed collecting data in an experiment intended to describe the behavior of the eye's pupil under well-controlled, defined, and dynamic lighting conditions during the performance of a standard cognitive task. The goals of this experiment included differentiating pupillary responses to changes in luminance from pupillary responses to changes in CW. This differentiation between the two types of pupillary responses is crucial for using the pupil as an indicator of CW since the pupillary responses to light changes can be several orders of magnitude greater than those to CW.

The present paper provides a detailed description of the methods and procedures used to collect the PS data. The paper also presents the complete raw dataset of a single individual volunteer to demonstrate the quality of the data. The presentation and visualization of such individual data are important since subsequent analyses present averaged data that obscures the characteristics of individual performance. The expectation is that this documentation will extend the usefulness of this dataset, which may be made available to qualified components of the wider research community.

Methods

Participants

Twenty-four volunteers of either gender between the ages of 19 to 45 participated in this study. None used bifocals or progressive focus lens corrections and none had intraocular lens implants. Furthermore, no participant had a self-reported history of epilepsy or flicker-induced seizure or migraine. No demographics, including gender and age, were recorded as part of this research.

This study was conducted under the U.S. Army Medical Research and Materiel Command, now the U.S. Army Medical Research and Development Command Institutional Review Board approved protocol #10751 and USAARL protocol #2018-010.

Equipment

Software.

The procedures were heavily automated to ensure consistent administration, and to allow the researchers to focus on monitoring data quality and participant performance. These procedures were executed using custom software developed in MATLAB (The Mathworks Inc., Natick, MA) and PsychToolbox (Brainard, 1997). The software scripts used for all stages of the procedure are described within the appropriate sections below, and are included in the Appendix E-H.

Visual display.

A 27-inch, high dynamic range, 4K liquid crystal display (LCD) monitor (ASUS PG27UQ) served as the sole light source for this study. During the development of visual stimuli, a USAARL-designed photocell system ensured that the monitor's temporal luminance output profile was sinusoidal at the appropriate oscillation rate and amplitude, and validated system set-up accuracy. A script, detailed in the Equipment Preparation and Calibration Procedure section below, daily checked the LCD monitor health and ensured the stability of the luminance settings and output. The display proved to be remarkably consistent across these daily checks and capable of illuminating the screen from 0.2 candelas per square meter (cd/m^2) to 110 cd/m^2 .

Eye tracking.

The EyeLink 1000 desktop-mounted, infrared, video-based eye tracking system (SR Research Ltd; Ottawa, Ontario, Canada), in its "Remote Mode" configuration, recorded the line of sight (LOS) and pupil area of both eyes, simultaneously, at a rate of 500 hertz (Hz) per eye, and without the need for head immobilization. Initialization of the calibration procedures as well as the starting and stopping of the recordings during the study were all automated using commands from the EyeLink MATLAB Software Development Kit (SR Research Ltd; Ottawa, Ontario, Canada) embedded in the custom MATLAB scripts. The scripts used to execute the procedures of the study (See Appendix E-H) require minimal input from the researcher during the experiment. EyeLink 1000 technical specifications are included in the operator's manuals

that are accessible via the manufacturer website: <http://www.sr-research.com/eb.html>.



Figure 1. SR Research EyeLink 1000 desktop system configured with the Remote Camera Upgrade. Photo Credit: SR Research, retrieved on 12 July 2021 via <https://www.sr-research.com/eyelink-1000-plus/>.

Equipment Preparation and Calibration Procedure

After the systems were turned on and stabilized, artificial eyes were positioned roughly where those of a participant would be located. A custom USAARL MATLAB script, included in Appendix F, automated daily equipment checks and camera calibration routines. The script first initialized control of the display to ensure that the monitor and computer communicated effectively. Next, the script initialized control of the eye-tracking system and prompted the researcher to focus the camera lens and ensure that the system effectively imaged the artificial eyes. A small data file was collected, and the collected pupil area measurements compared with the known pupil area of the artificial eyes. These steps ensured that the eye tracking system was able to record data and confirmed that the components had not been moved between testing sessions. The software then stepped the display monitor through each of the three specified mean luminance levels, and prompted the researcher to measure the luminance with a Minolta LS-100 (Konica-Minolta Co. Ltd., Tokyo, Japan). These luminance measurements were recorded and checked with the intended values of 0.8, 8, and 80 cd/m^2 to ensure the monitor's settings had not altered, and to ensure proper screen function. Only after completing all these calibrations within specifications, was the system cleared for daily data collection. It should be noted that the laboratory space and equipment were devoted exclusively to the study for the duration of the data collection period, thus the recorded values from these calibration checks never deviated from specifications. The MATLAB code to execute these control and calibrations is included in Appendix F.

Experimental Task Breakdown

Light stimulation procedure.

Lighting Factor A.

Mean luminance of the computer monitor set at one of three possible levels:

1. Dim: About 0.8 cd/m², which is a low photopic luminance in which the pupil is expected to be relatively large, on average about 7 millimeters (mm) in diameter.
2. Moderate: About 8.0 cd/m², which is a moderate photopic luminance and the pupil is expected to be intermediate, on average about 4 mm in diameter.
3. Bright: About 80 cd/m², which is typical indoor office luminance or what one encounters out of doors on a cloudy day, resulting in a pupil of nearly 2.0 mm in diameter.

Lighting Factor B.

The mean luminance (Lighting Factor A) modulated $\pm 25\%$ at one of four temporal frequencies (Lighting Factor B). The $\pm 25\%$ modulation was selected since the literature suggests that the noise in the pupil's response to this level of modulation is minimal (Barrionuevo et al., 2014; Myers et al., 1993). The four temporal modulation frequencies were:

1. 1.0 cycle per second (Hz): This modulation frequency is near the peak of the pupil's change frequency (the pupil cannot follow modulations much faster than this frequency).
2. 0.2 Hz: The pupil should be able to follow this low frequency but it is expected to fall off dependent upon luminance level.
3. 0.1 Hz: The pupil should be able to follow this low frequency but there may be an additional fall off dependent upon luminance.
4. 0 Hz: Control condition; luminance does not modulate.

Thus, there are 12 different lighting conditions and all participants were tested with all of them.

Cognitive task (n-back).

The study design used an auditory form of the Delayed Digit Recall Task, or n-back task to challenge cognitive workload. The n-back task is a widely used, continuous performance task intended to manipulate mental workload and working memory [see (Owen et al., 2005) for a meta-analysis]. For the current study, a recorded voice presented 20 single digits one-by-one selected from 1 through 9, in a predetermined random sequence. A presentation rate of one digit every 1.5 seconds allowed participants enough time to encode the digit heard and provide their spoken response. The assigned delay between digit presentation and digit response determined task difficulty. The minimum difficulty was the immediate identification of the digit upon presentation, the so-called n-back 0 condition. More difficult was the n-back 1 condition in which digit-1 is presented and held in memory, but not reported until the digit-2 is presented. Upon presentation of digit-2, the participant reports digit-1 while holding digit-2 in memory for subsequent recall following presentation of digit-3. Table 1 illustrates the n-back 0 through n-

back 4 conditions. A digital recorder captured and retained responses for post-hoc scoring. Based on the temporal presentation of 1.5 seconds per digit for 20 digits, an individual n-back trial of 20 digits required 30 seconds to complete.

Table 1. Levels of N-Back Difficulty

Level	Digits Heard	Digit Recalled & Stated	Digits Maintained in Memory
0	1	1	Not Applicable (N/A)
1	8, 7	8	7
2	1, 9, 4	1	9, 4
3	6, 5, 8, 1	6	5, 8, 1
4	3, 7, 5, 4, 2	3	7, 5, 4, 2

Table 1 Notes:

0-back: in which the participant reports the digit just heard.

1-back: in which the participant reports the prior digit while storing in memory the immediate digit for subsequent recall.

2-back: in which the participant recalls and reports the digit that preceded the prior digit, while holding in memory the prior digit and storing in memory the immediate digit.

3-back: in which the participant recalls and reports the digit that is three digits back while storing three digits in memory.

Experimental Session Breakdown

Participant instruction and eye tracking calibration.

Upon arrival, a prospective volunteer was guided through the consent process in a private setting. Individuals volunteered for the study after they read the informed consent document, had all their questions answered, and signed the consent document. Following informed consent, the volunteer was seated in the recording suite and guided through an information packet (See Appendixes B and G). After the participant acknowledged understanding the procedure and the task, a testing and practice function was run from within MATLAB using code provided in Appendix G. The ambient room lights were extinguished and a three sided, light baffling shroud erected around the monitor, extending past the participant's head, so that the only visible light reaching the participant came from the monitor with some stray visible light from the eye tracking system's infrared emitter. The script then initialized and verified the connection to the monitor screen and the eye-tracking system. Once these checks cleared with no issues, the participant completed a 9-point (x,y) calibration procedure on the monitor. Following the 9-point calibration, the participant fixated the center of the monitor and the monitor stepped to each mean luminance level, holding each luminance for 60 seconds. The participant's LOS and pupil area were captured during the last 30 seconds for each luminance step. If the participant did not maintain fixation centered on the screen, the participant was reminded of the necessity of maintaining fixation on the fixation target, and this stage of the procedure repeated. Once the procedure was completed successfully, the participant's average pupil area was calculated and recorded for each luminance step, and then displayed to the experimenter. If there was less than a 1 mm difference in the participants' pupil areas of either eye between each of the luminance steps, or a difference of greater than 1 mm between the areas of the eyes at each of the steps, the

session was terminated and that participant was excluded from analysis. Of the 24 participants, one was excluded because the camera system was unable to detect effectively the pupils, and two additional participants were excluded because their pupils did not change sufficiently in response to the luminance changes.

Once the participant cleared these calibration and validation steps, the participant had three practice trials at each of 0-, 1-, 2-, and 3-back. The researcher scored the final practice trials at 1-, 2-, and 3-back tasks. To ensure the tasks would elicit consistent and sufficient challenges across task difficulty, participants were excluded from participation if they failed to achieve a 50% correct response rate in the 1-back trial, or performed better than 50% correct on the 3-back trial. One participant was excluded for the former and one for the latter reason. Participants who performed better than 50% on the 2-back, were administered the 3-back condition in its place to serve as the most difficult challenge tier. Six of 19 participants included in the analysis were administered the 3-back instead of the 2-back.

After completion of the initial calibration, validation, and training procedures, the participant was allowed to take a break and ask any final questions before starting the experiment proper.

Experimental session and data collection procedure.

Once the participant was ready to begin the experiment, the individual assumed the proper position in the recording suite. The script included in Appendix H was run, and the participant's gaze position was again calibrated until the estimated LOS accuracy did not exceed two visual degrees. An eye-tracking status display on the researcher workstation enabled the researcher to monitor the participant's eyes during the entire study in real time to ensure that the eye tracker was successfully collecting the eye data and that the participant was performing the fixation task as instructed.

During all the experimental tasks, each participant was instructed to look straight ahead into the homogeneously illuminated screen while keeping the LOS aligned toward a stationary, one visual degree, red fixation cross. Participants were not instructed to avoid blinking. In fact, researchers made no mention of blinking to avoid altering the participants' natural blink-rates, which are important for subsequent analysis.

All trials at a given mean luminance level were administered within a single administration block. A five-minute adaptation period preceded each luminance block during which the luminance remained constant at the block's mean luminance. This five-minute period allowed the visual system to adapt and the pupil's response stabilize to that luminance. Prerecorded excerpts from a randomly selected, public domain, audiobook were presented during this time to prevent excessive boredom and fatigue.

Trials for each of the 3 n-back challenge levels were administered during presentations of each of the 4 luminance oscillation rates for a total of 12 trials in each mean luminance block. We administered these 12 trials in a pseudo-randomized order for each of the 3 mean luminance levels (0.8 cd/m^2 , 8 cd/m^2 , 80 cd/m^2), for a total of 36 trials in the entire experiment for each participant. The trial order and block order were randomized; furthermore, each participant experienced a different random order following a random Latin-square manipulation such that

each trial type occurred at each sequence position across all participants. Time and protocol constraints prevented the replacement of the five excluded participants, so this balancing procedure was not perfectly implemented in this data set.

The 0-back, 1-back, and either 2-back or 3-back provided the CW challenge conditions. As described above, the n-back task consists of recorded verbal presentations of a string of 20 single digits at a rate of 1 digit every 1.5 seconds. Thus, the procedure involved the performance of a cognitive n-back task for 30 seconds while fixating a homogenously illuminated screen temporally modulating in brightness at a frequency of 1.0 Hz, 0.2 Hz, 0.1 Hz, and 0.0 Hz (no modulation). Thus the screen oscillated in a sinusoidal fashion 30 times in the 1 Hz condition. The screen brightness oscillated six cycles during the 0.2 Hz trials, and oscillated 3 cycles in the 0.1 Hz frequency conditions. For the 0.0 Hz frequency, the screen remained constant at the mean luminance with no modulation (see Figure 2). Between each trial, the participant viewed a luminance-matched blank screen. This screen also displayed written instructions that included the n-back level of the next trial. After five seconds, the participant was asked to remain in position, and prompted to press a key to initiate the next trial. Longer breaks were given between the larger blocks. If a participant elected to get up or leave the recording suite, for any reason, calibrations were repeated before beginning the next trial. Note that no participant elected to leave the room between trials and that most participants did not pause for longer than the five-second minimum break between trials. At the completion of testing for each luminance, participants reported the subjective workload posed by each of the tested n-back levels using the NASA-TLX (Appendix B).

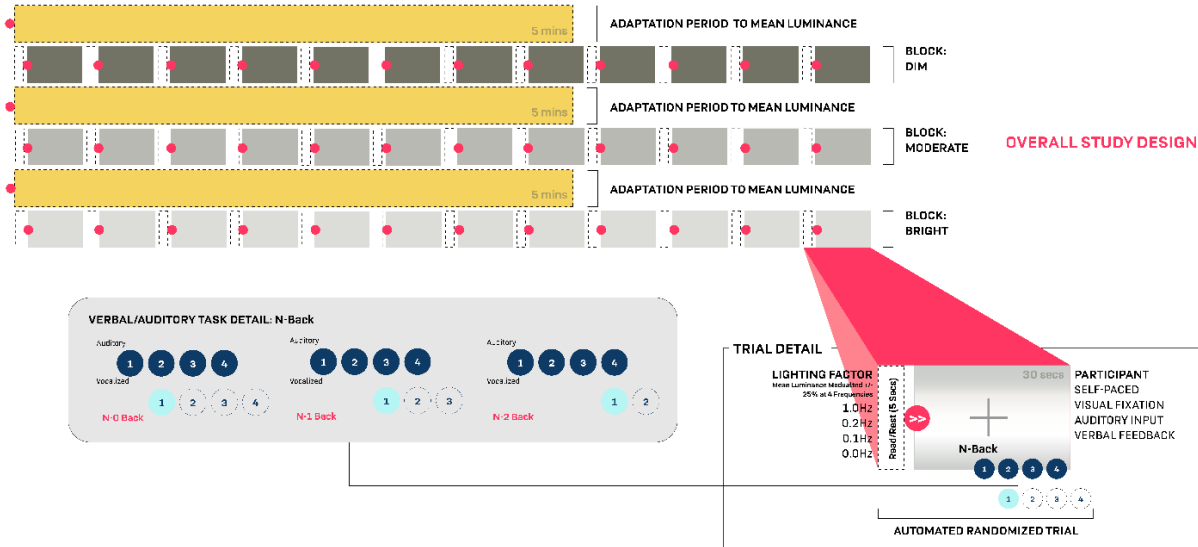


Figure 2. Overall study design.

Data storage.

Eye tracking data (LOS, PS) were collected by the EyeLink Host personal computer (PC), and then copied to the Display PC after each trial using commands within the MATLAB scripts running the experiment. These data were combined with pre-trial calibration information,

task parameters, and event timestamps. At the conclusion of each trial block these data were stored within a single MATLAB data structure for later analysis. Recordings of the participants' verbal responses were transferred to the display PC after the conclusion of each experiment, and stored in the same folder as the MATLAB data files, but were left as a separate file and not integrated.

Results

To illustrate the kind and quality of the PS data recorded during this experiment, the raw PS data traces obtained from the left eye of a single subject are presented as a function of time. These time series data are the entire 30 second traces obtained at 500 Hz. Data from the right eye were collected simultaneously, also at 500 Hz. Only one eye is presented here for the sake of parsimony. On visual inspection, the data from the right eye are essentially identical. Figure 3 displays PS during the 0-back conditions. Similarly, Figure 4 and 5 display PS measures obtained during the 1-back and 2 (or 3)-back conditions, respectively.

Figures 3, 4, and 5 each display four graphs. All these dozen graphs are scaled identically to facilitate direct visual comparisons. The abscissa plots time in seconds, spanning the full 30 seconds of each trial. All the 15000 data points collected over the 30-second testing duration are plotted. The ordinate is the PS in pixels, which represents an absolute linear metric. Calibrations established that 5 mm pupil diameter was 838 pixels at a viewing distance of 50 centimeters (cm) from the camera.

Each graph contains three tracings, one for each of the three mean luminance values used. The vertical separation between the 3 tracings is the difference in PS resulting from the mean luminance differences such that the brightest screen luminance (80 cd/m^2) produced the smallest PS and the dimmest screen luminance (0.80 cd/m^2) produced the largest PS. Thus, the vertical separation of the three tracings is not arbitrary, but is the direct result of the stimulus conditions. This result is consistent and obvious in all dozen graphs.

In Figures 3 through 5, the upper left quadrant presents PS recorded over the trials during which the homogeneously illuminated display had a temporal modulation frequency of 0 Hz. The upper right quadrant presents PS recorded over the trials during which the homogeneously illuminated display had a temporal modulation frequency of 0.10 Hz. The lower left quadrant presents PS recorded over the trials during which the homogeneously illuminated display had a temporal modulation frequency of 0.20 Hz. Lastly, the lower right quadrant presents PS recorded over the trials during which the homogeneously illuminated display had a temporal modulation frequency of 1.0 Hz.

In Figure 3, the upper left quadrant shows the 30 second traces of PS of the eye passively viewing the steady, homogenous field for the 0-back task. This is essentially the control condition showing the expected PS fluctuation and variability at each of the luminance levels. Thus, there is no expected fluctuation in PS associated with changes in luminance. Slight transient dips in PS, at about 11, 16, and 25 seconds, while unexplained, may reflect suppressed changes in fixation causing a change in pupil diameter. Despite these minor transients, the PS tracings are essentially flat over the 30 seconds of the data collection interval. It should be noted that the difference in PS between the bright and middle luminance conditions is about half that of

the PS difference between the middle and dim luminance conditions despite the fact that the difference in luminance between the dim and middle and the middle and bright luminance conditions is essentially the same, a factor of ten. Notably, the difference in mean PS is essentially the same for the four quadrants.

The 0.1 Hz luminance modulation is equivalent to 1 cycle per 10 seconds or 3 cycles over the testing interval of 30 seconds. The upper right quadrant of Figure 3 shows the PS following this stimulus. The three cycles are most evident in the middle luminance but less evident at the bright luminance where the peak trough size difference is noticeably less, again indicating a response compression with increased luminance. At the dim luminance, there is some indication of an entrained, cyclic PS response, but it is much attenuated and at best noisy, which is exactly as expected with a near threshold stimulus.

The lower left quadrant of Figure 3 shows the following pattern of the PS to the 0.2 Hz modulation, which is 2 cycles per 10 seconds and equivalent to 6 cycles per 30 seconds. The six cycles of peak and trough PS is evident in the data for the high and middle luminance tracings but seem to be lost in PS variability in the low luminance tracking. The mean PS seems to be a constant over the 30 seconds of tracking. A brief interval of lost data seems to be evident half way through the tracing for the bright luminance condition. The peak/trough difference is noteworthy. This difference in PS is visibly larger under the middle luminance condition than under the bright luminance condition, a difference that may be due to the dynamics of the pupil's response. A similar but much reduced type of response compression may be seen in the comparison between the moderate and bright luminance conditions at the 1 Hz modulation rate in the lower right quadrant. It should be noted that although the PS response to the dim luminance condition is obscured at 0.2 Hz, it is evident at 1.0 Hz.

The lower right quadrant of Figure 3 shows the most striking, obvious PS response pattern. Here, the homogeneous display modulated at 1 Hz, and counting peaks or troughs in PS shows the coupled 1 Hz response, suggesting that the pupil might be considered little more than a photocell. This PS behavior is evident in each of the three tracings. The average diameter seems to be approximately constant for the duration of the 30 seconds for the 80 and 8 cd/m^2 tracings; but for the 0.8 cd/m^2 tracking a slight discontinuity is apparent at about halfway through the trial which might be due to a slight change in fixation. At such a low luminance, a slight perturbation can be expected to have a large impact in a linear measurement. Furthermore, the large change in PS at the start of the trial is almost certainly an artifact of the transition to the beginning of the trial and should be treated as such in the analysis and interpretation of these types of data.

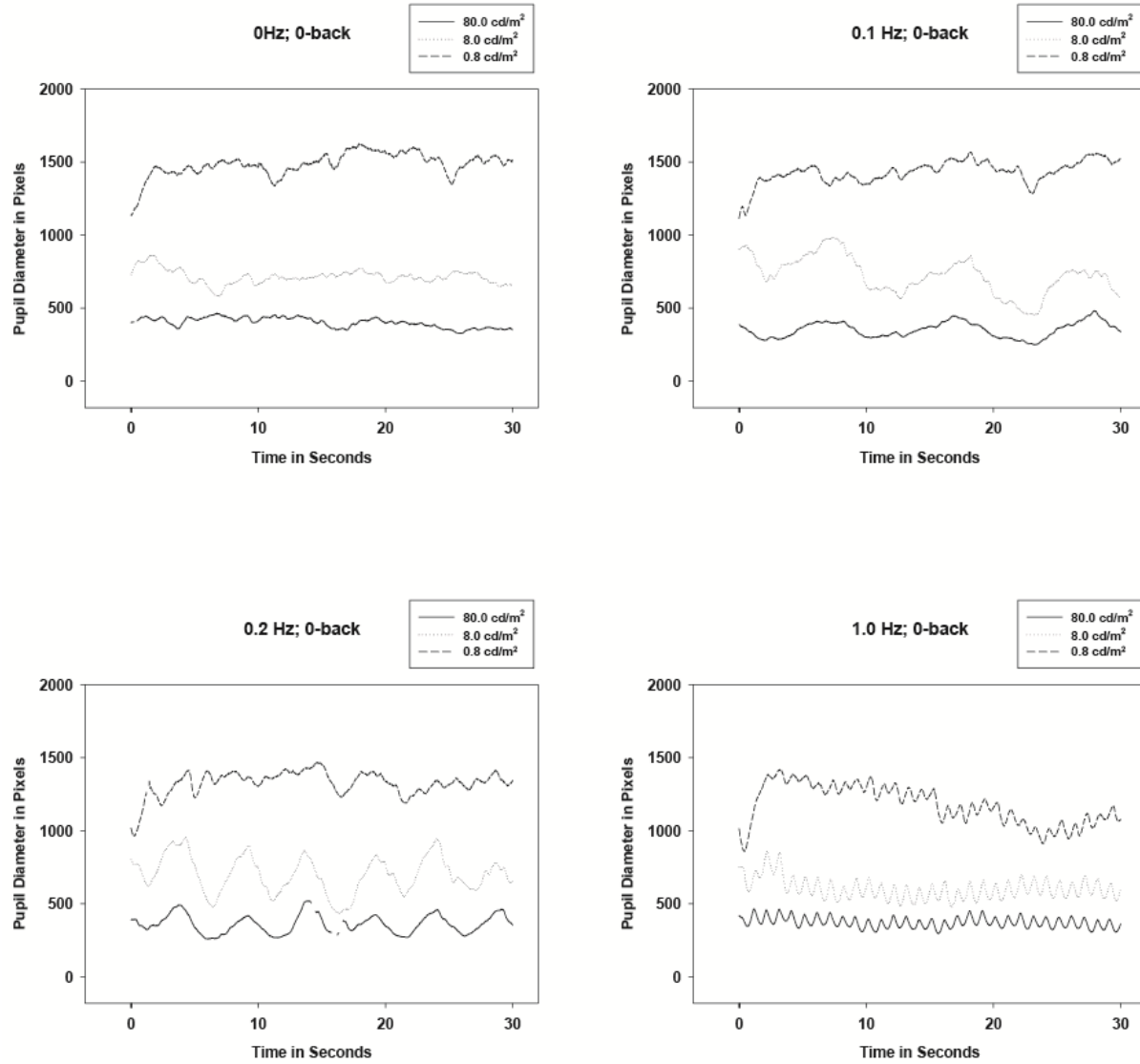


Figure 3. The pupil diameter in pixels as a function of time in seconds recorded at 500 Hz in the left eye of a single subject during the 0-back task. Data displayed in the upper left quadrant were collected with a 0.0 Hz. Data displayed in the upper right quadrant were collected with a 0.1 Hz display screen temporal modulation of $\pm 25\%$ lumance. Data displayed in the lower left quadrant were collected with a 0.2 Hz display screen temporal modulation of $\pm 25\%$ lumance. Data displayed in the lower right quadrant were collected with a 1.0 Hz display screen temporal modulation of $\pm 25\%$ lumance. Each panel displays three data tracings, one for each of the three mean screen luminance values identified in the figure legend.

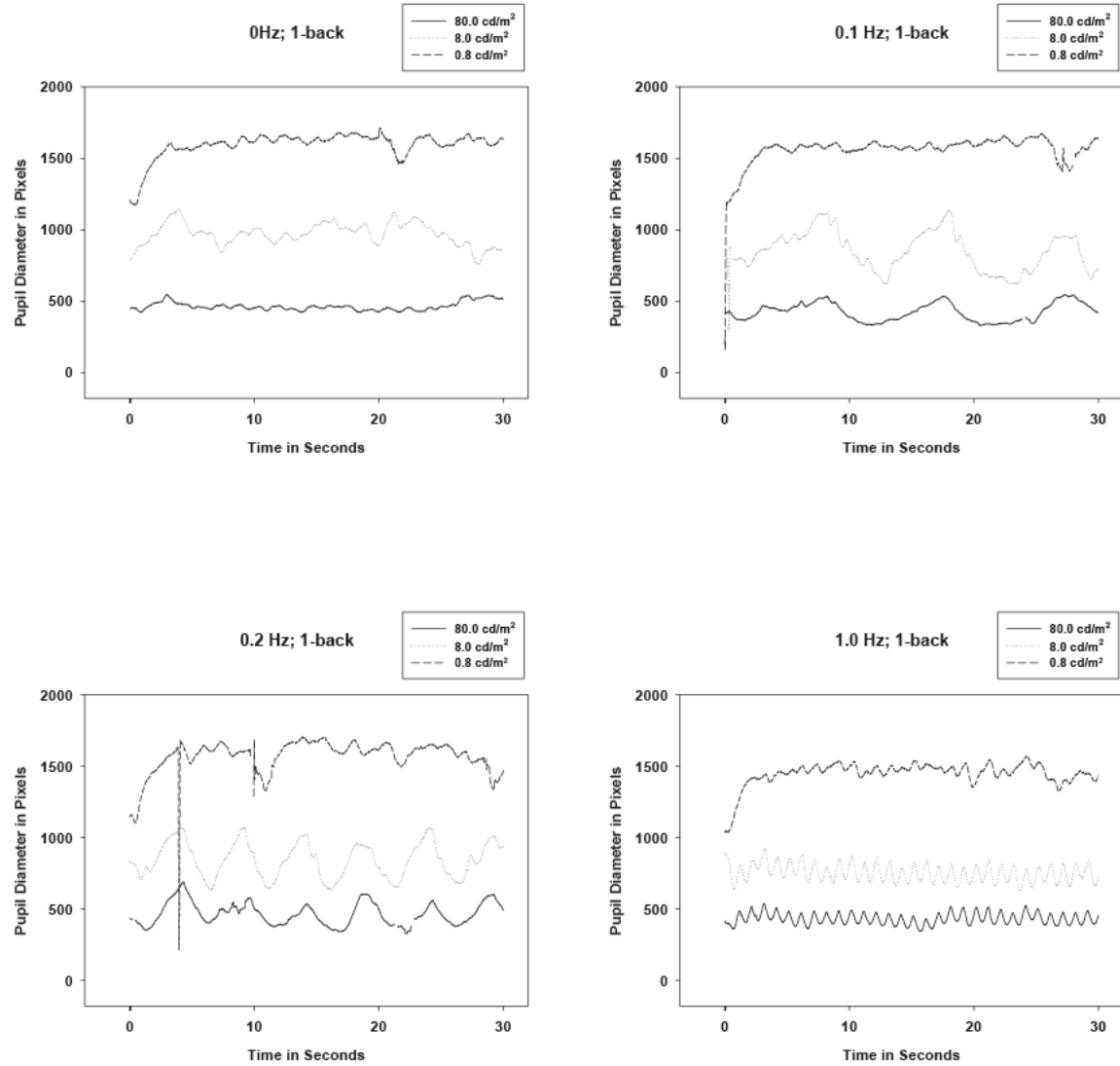


Figure 4. The pupil diameter in pixels as a function of time in seconds recorded at 500 Hz in the left eye of a single subject during the 1-back task

The exact same patterns of PS as a function of luminance conditions described above for Figure 3 apply for Figure 4 with some minor difference. Specifically, the lower left quadrant of Figure 4 shows a transient loss of data at about the fourth second of data collection under the dim condition and a similar transient partial loss of data at about the ninth second. These transient data lapses appear as spikes toward zero in the data trace. Similar spikes are evident at the beginning of data collection in the upper right quadrant.

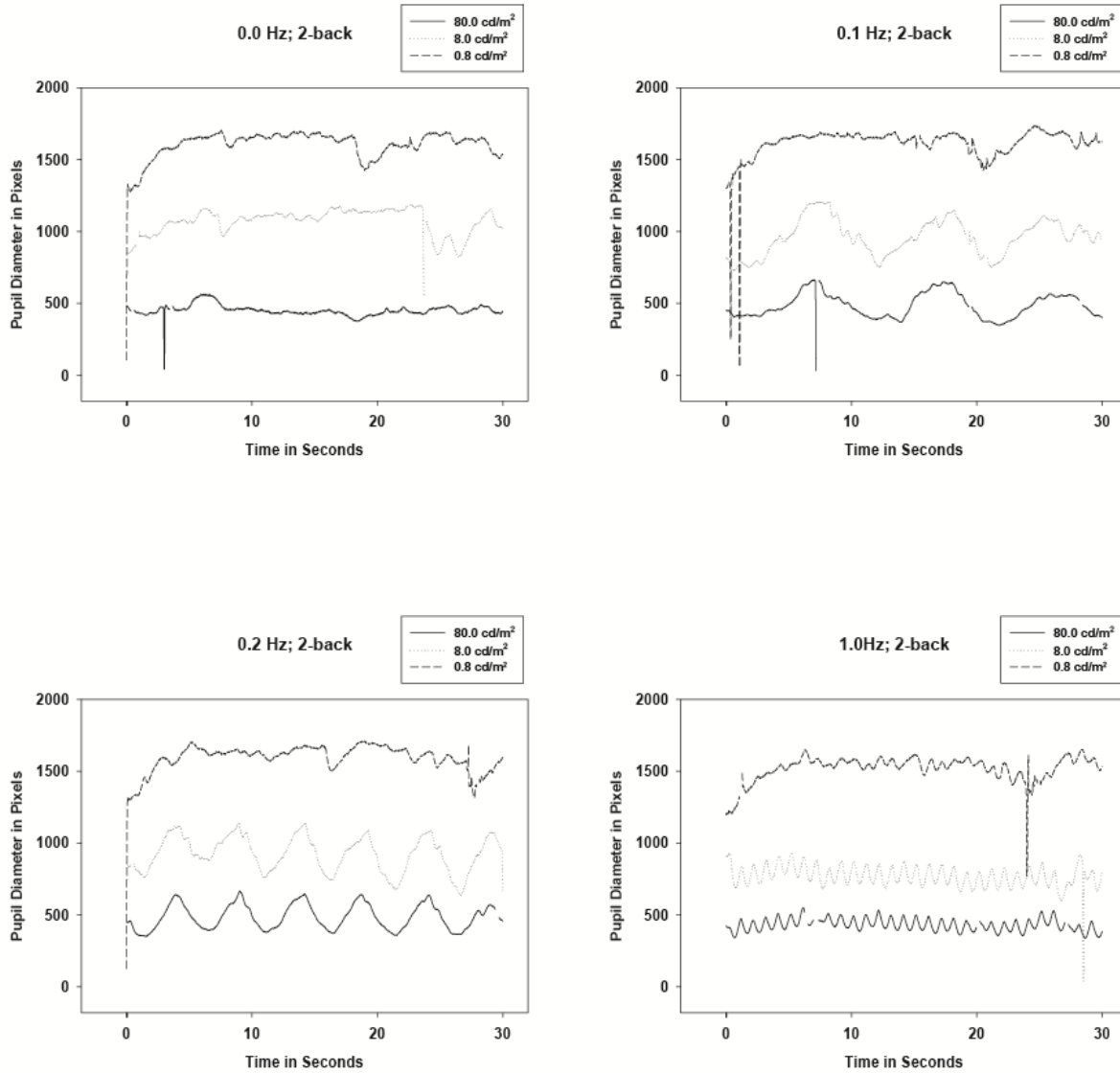


Figure 5. The pupil diameter in pixels as a function of time in seconds recorded at 500 Hz in the left eye of a single subject during the 2- or 3-back task .

The PS data collected during the 2 (or 3)-back task shown in Figure 5 are essentially indistinguishable from the data displayed in Figures 3 and 4. One intriguing difference is the increase in the incidence of data lost with increased workload tasking. This increase may represent a blink, and blinks are known to increase with cognitive workload. However, a more careful analysis of these data would be required to evaluate this surmise.

Discussion and Recommendations

The present report is one of several documenting different aspects of an experiment designed to evaluate the feasibility of harnessing changes in PS as a physiologically based objective estimate of an individual's cognitive engagement with a challenging task. In essence, the goal is to use PS as a tool to monitor, if not measure, CW in operational environments. The data presented here are discussed as a prerequisite control condition and demonstration, preparatory to the CW components. The discussion emphasizes the accuracy, precision, and temporal resolution with which the PS recordings were made in response to the dynamic luminance changes and their synchronization with the n-back cognitive task. The discussion did not address the impact on PS of CW or arousal or the engagement with the cognitive task. Rather, the report addresses the impact of the luminance manipulations, conditions that drove the PS in the expected fashion. An essential pre-condition of the study is the demonstration that the effects of specific luminance parameters are evident in the behavior of the pupil. Without these effects, the study would be highly suspicious. The presence of these effects demonstrate the precision of the measurements. Moreover, it is important to note these luminance conditions drove the PS essentially over its full dynamic response range, from minimal response to saturation. Thus, the luminance stimulus conditions successfully controlled the PS, thereby providing a dynamic but predictable and stable baseline with which to assess the ability of the PS to encode changes associated with cognitive task performance in the presence of such dynamic lighting conditions.

The literature contains several reports of PS as an 'index of cognitive activation' or as a 'task evoked pupillary response.' Such reports have relied on extensive analysis using methods that are either proprietary or computationally demanding. Much of this work has been focused on evaluating the cognitive components of the PS response under controlled and relatively stable lighting conditions. While some operational environments, such as command and control centers, or remotely piloted vehicle workstations, potentially may be configured to provide well-controlled lighting conditions; other operational environments may not. One remaining challenge is to assess the strengths, weaknesses, and potential utility of pupillary behavior as an index or sign of cognitive engagement under a range of lighting conditions. For this, an understanding of the pupil's response to dynamic changes in light as well as to cognitive engagement and their interactions is essential. The present dataset was collected to help assess these issues. To that end, the study manipulated the lighting conditions simultaneously with cognitive workload demands to assess how they interacted. While the mean luminance conditions drove the PS over its complete dynamic response range, the continuous temporal sinusoidal luminance changes ranged from one cycle per second to one cycle per ten seconds.

Despite these rather slow and limited conditions, they did provide important information that helps define the challenge and the range of possible solutions for using PS to monitor cognitive engagement. For example, the change in PS between the dim and the middle luminance was greater than between the middle and bright luminance. This suggests a response compression in PS which may limit the luminance range over which the PS can provide useful information. The other source of PS response compression is the luminance modulation frequency. The PS amplitude modulation with a 1.0 Hz stimulus is a fraction of the PS amplitude modulation at the lower frequencies.

Possibly the most important single observation is that the lighting conditions overwhelmingly determined pupillary behavior. In other words, the several effects of the different illuminations conditions are clearly apparent in the Figures 3 through 5. In fact, there really is no need for statistical inference. Within each figure, the reader can identify immediately the differences among the luminance conditions. Furthermore, the differences among the four panels of each figure are obvious and obviously consistent between Figures 3, 4, and 5. Equally obvious is the fact that there is no clear evidence of any differences among 0-back, 1-back, and 2 (or 3)-back: The luminance differences in PS dwarf the PS difference due to cognition. In fact, based on the inspection of Figures 3 through 5, it is reasonable to question whether there are any systematic changes in the PS attributable to CW. On the other hand, such small but systematic differences in PS associated with CW may well become visible by increasing the magnification of the ordinate. The ordinate is compressed in the figures presented here to simplify the presentation of the sample database.

The data presented here show periods of data loss, questionable discontinuities, and baseline shifts. While these data interruptions and dropouts are relatively infrequent, their effects can be substantial. Thus, these data, as clear and systematic as they are, demonstrate that the individual data traces need to be inspected and filtered to minimize the impact of these spurious influences without distorting the veritable data. Several of the traces evidence a systematic dilation of the pupil at the initiation of the trial. This is particularly the case under dim illumination. Discovering and documenting such differences will be provided in the subsequent papers that address these data.

References

- Aura, C., Temme, L., McAtee, A., O'Brien, K., Hayes, A., Bowers, B., Baugher, K., & St Onge, P. (2021). Evaluating Pupillometric Indicators of Cognitive Workload and Autonomic Tone in Changing Luminance Conditions. Military Health System Research Symposium, Kissimmee, FL.
- Aura, C., Temme, L., St. Onge, P., McAtee, A., Wilson, M., Bowers, B., Baugher, K., & Hayes, A. (2021). Operator state monitoring via pupillometry: measuring mental workload under varying luminance conditions. SPIE Defense + Commercial Sensing.
- Barrionuevo, P. A., Nicandro, N., McAnany, J. J., Zele, A. J., Gamlin, P., & Cao, D. (2014). Assessing rod, cone, and melanopsin contributions to human pupil flicker responses [Original Paper]. *Investigative Ophthalmology & Visual Science*, 55(2), 719-727. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC24408974/?tool=EBI>
- Brainard, D. H. (1997). The Psychophysics Toolbox. *Spatial Visualization*, 10(4), 433-436.
- Hancock, P. (2006). Mitigating Stress, Workload, and Fatigue on the Electronic Battlefield [Final Report]. *DTIC*, 1-2905.
- Myers, G. A., Gannon, J. A., & Stark, L. W. (1993). Level dependent signal flow in the light pupil reflex. II. Phase velocity of responses to sinusoidal light stimuli. [Original paper]. *Biological Cybernetics*, 68(3), 235-240. <https://www.ncbi.nlm.nih.gov/pubmed/8452892>
- Owen, A. M., McMillan, K. M., Laird, A. R., & Bullmore, E. (2005). N-back working memory paradigm: A meta-analysis of normative functional neuroimaging studies [Meta-analysis]. *Human brain mapping*, 25(1), 46-59.

Appendix A. Acronyms and Abbreviations

ANS	autonomic nervous system
cd/m ²	candelas per square meter
COTS	commercial-of-the-shelf
CW	cognitive workload
Hz	Hertz
LOS	line of sight
NASA	National Aeronautics and Space Administration
NASA TLX	NASA Task Load Index
PLR	pupillary light reflex
PS	pupil size
PSN	parasympathetic pathway
USAARL	U.S. Army Aeromedical Research Laboratory

Experiment Instructions

As implemented in the DVE Pupil-Luminance Study

THANK YOU FOR VOLUNTEERING TO PARTICIPATE!!

OVERALL TASK ORDER

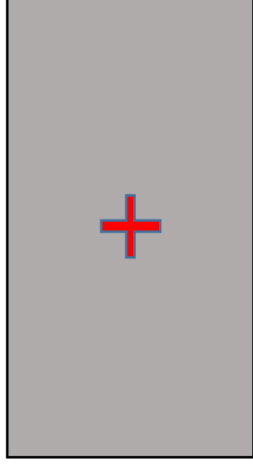
1. Introduction and Signing Consent Form
2. Review this instruction packet.
3. Successfully complete the Calibration and N-# Back Practice
4. Break to ask questions, use restroom, stretch, etc.
5. Start the Experiments!
 1. Calibration Task
 2. Light Level 1 Blocks
 1. Adaptation Period and Story Listening (5 Mins)
 2. N-0, N-1, and N-2 Trials (12 – 30 Seconds Each)
 3. Light Level 2 Blocks
 1. Adaptation Period and Story Listening (5 Mins)
 2. N-0, N-1, and N-2 Trials (12 – 30 Seconds Each)
 4. Light Level 3 Blocks
 1. Adaptation Period and Story Listening (5 Mins)
 2. N-0, N-1, and N-2 Trials (12 – 30 Seconds Each)
6. Debrief and Parting Questions

CALIBRATION TASK

- Get into a comfortable posture that you can maintain, and look at the center of the screen.
 - Your experimenter may need to adjust the camera during this time, try to stay still while they work around you.
- Once the camera is ready, a small white circle will appear on the screen.
- Look at it's very center and do not look away.
- Once the computer logs your eye position (this may take a few seconds), it will move to a new location.
- Find it and stare at it's center until it moves again.
- It will move to several positions on the screen before completing
- Once complete, you may be asked to repeat the task to improve the calibration.
- IT IS MOST IMPORTANT THAT YOU DO NOT LOOK AWAY FROM THE CENTER OF THE WHITE CIRCLE WHEN IT IS VISIBLE ON THE SCREEN IN FRONT OF YOU.
 - Note: Everyone has veins in their eyes, and sometimes these veins are positioned just so that the camera system gets confused and is unable to track your eye position. If this is the case, the experimenter will let you know and you will not be asked to continue participating in the study
 - IF WE ARE UNABLE TO CALIBRATE, THIS DOES NOT MEAN THERE IS ANYTHING WRONG WITH YOU OR THAT THERE IS ANYTHING WRONG WITH YOUR EYES.
 - If you have concerns about your eyes or visual health, please see an optometrist, we are not trained or authorized to help give advice regarding those concerns.

ADAPTATION PERIOD INSTRUCTIONS

- The adaptation period is there to give your eyes time to adjust to the light level for that block of experiments.
- An audio-book will play for you to listen to.
- Keep your head up and keep looking at the cross in the center of the screen until the trial is over.
- After the adaptation period, you will begin the N-# back trials.



N-# BACK TASK INSTRUCTIONS

- Keep looking at the cross while you perform the task
- During this task you will hear a sequence of numbers spoken through the speakers in front of you.
- You will give your responses in the 1 ½ second pause between each of the spoken numbers.
- Your response will be to speak whichever number you heard as many numbers (#) back from the last number spoken indicated by the N-# back line on the information screen before each trial.
- If you make a mistake, miss a number, or miss the response window, keep going.
- Once started, the task cannot be stopped, and trials cannot be repeated.
- Some of the trials will be very challenging, they are designed to be difficult, and it is most important that you keep trying and give your best effort to get as many responses correct as possible.
- TO BEGIN EACH TRIAL, READ THE FIRST TWO LINES OF THE INSTRUCTIONS
 - For example: " Trial 12 out of 36, N-2 Back"

N-0 BACK TASKS

N-0 Back means to speak (or repeat) the last number that you heard spoken after each number you hear

This is the 1 ½ Sec. Response Window

You Hear: 1	2	3	4	5	6
You say:	1	2	3	4	5
					6

This is the correct response for this window for this type of N-0 Back task

You Hear: 3	6	9	7	1	2
You say:	3	6	9	7	1
					2

Remember: Speak loud and clear, and keep your eyes looking at the cross in the center of the screen while you are listening to the numbers and speaking your responses.

N-1 BACK TASKS

N-1 Back means you respond with the number that was spoken 1 number before, or 1 back in the order of numbers you hear

-- means you don't say anything in that response window

You Hear: 1	2	3	4	5	6
You say: --	1	2	3	4	5

There is no N-1 back number in the first response window, so you wait until the second response window to say the first number you heard

You Hear: 3	6	9	7	1	2
You say: --	3	6	9	7	1

You continue to say the number 1 back in the order from the last number you heard in every window after that, until the sequence is over.

You must get at least 25% of the numbers correct in this condition to participate, so keep trying!

N-2 BACK TASKS

N-2 Back means you will say the number spoken two before (or 2 back from) the last number you heard.

You Hear: 1		2	3	4	5	6
You say:	--	--	1	2	3	4

You will wait 2 response windows before saying the first number in the sequence, and then give a response in every window after that

You Hear: 3		6	9	7	1	2
You say:	--	--	3	6	9	7

THIS CONDITION IS VERY DIFFICULT, BUT IT IS CRITICAL THAT YOU KEEP TRYING UNTIL THE END
 If you make a mistake, KEEP GOING, focus trying to say the correct number in as many windows as possible
 Your score is based on how many you get correct, not how many you get correct in-a-row.

N-3 BACK TASKS – ONLY IN THE PRACTICE

N-3 Back means you will say the number spoken two before (or 3 back from) the last number you heard.

You Hear: 1		2		3		4		5		6
You say:	--		--		--		1		2	3



You will wait 3 response windows before saying the first number in the sequence, and then give a response in every window after that

You Hear: 3		6		9		7		1		2
You say:	--		--		--		3		6	9

THIS VERSION IS EXTREMELY DIFFICULT, YOU WILL MAKE MANY MISTAKES

If you make a mistake, KEEP GOING, focus trying to say the correct number in as many windows as possible. If you can get more than 25% of the numbers correct (That is only 5 numbers!) you may qualify for advanced testing and your session will be complete for today!

DO YOUR BEST AND DON'T STOP OR QUIT

The tasks, especially the N-2 Back and N-3 Back, are designed to be very hard.
It is MOST IMPORTANT that you continue to work hard and try to give as many of the correct answers that you can.

You Hear: 1	2	3	4	5	6
You say:	1	2	6	4	5
					6

If you forget the right answer or accidentally say a wrong answer, do not stop!
Do your best to give the right answer for the next window.

You Hear: 3	6	9	7	1	2	6	1
You say:	--	--	3	2	--	7	1
							2

If you make several mistakes in a row, do your best to get back on track as soon as you can.

Focus trying to say the correct number in as many windows as you can.

FINAL NOTES BEFORE WE START

- You will now have a chance to see what the screens will look like and an opportunity to practice the N-# Back tasks.
- If you are unable to successfully complete the calibration or the N-# Back tasks, you may not be able to continue participating in the study.
- The main set of experiments will not include the N-3 Back task.
- If you need to use the restroom, now is the time to go.
- If you have any questions about the tasks, please ask your experimenter.
- Once the experiment begins, we want to maintain a consistent and quick pace, so try to avoid conversation and any unnecessary interruptions.
- Make sure you are sitting in a comfortable position and posture that you will be able to maintain until the end of the experiment.
- If you wish to opt-out, or stop participating for any reason, you are welcome to do so
- The entire session will take around 2 hours.
- **THE AMOUNT OF LIGHT YOUR EYES SEE IS VERY IMPORTANT. DO NOT CHECK YOUR PHONE BETWEEN TRIALS**
 - IF YOU NEED TO CHECK YOUR PHONE, TELL YOUR EXPERIMENTER AND THEY WILL LET YOU KNOW WHEN THAT IS OK

Appendix C. Experimenter Checklist

EXPERIMENTER CHECKLIST

DVE PUPIL

V1 – 07012019

BEFORE SUBJECT ARRIVES (only necessary before 1st Subject of Day):

- ☐ EYELINK HOST AND DISPLAY COMPUTERS ON 1 HOUR PRIOR TO START
- ☐ RUN MATLAB ON DISPLAY COMPUTER
- ☐ PLACE CHINREST WITH PUPIL TEST CARD IN CENTER OF SUBJECT TABLE
- ☐ OPEN PreSessionCheck.m
- ☐ ENTER SUBJECT NUM INTO pre.subjectNum
- ☐ CLICK RUN AT TOP OF WINDOW (Green Play Button)
- ☐ HOLD LUMINANCE METER NEXT TO PUPIL CARD AND AIM AT SCREEN
- ☐ PULL TRIGGER AND GET VALUES FOR EACH OF THE THREE SCREENS
- ☐ ENSURE PROGRAM FINISHES CORRECTLY AND THERE IS NO RED TEXT

AFTER SUBJECT ARRIVES:

- ☐ GREET, HAVE SIGN CONSENT FORM
- ☐ HAVE THEM SIT IN SUBJECT SEAT AND FIND A COMFORTABLE POSTURE
- ☐ START RECORDER, STATE THE DATE AND SUBJECT NUMBER (NO NAMES!!) AND PLACE ON TABLE IN FRONT OF YOU
- ☐ ASK SUBJECT TO REVIEW THE INSTRUCTION SHEET AND ANSWER ANY QUESTIONS
- ☐ OPEN LumDynPupilPracV1.m
- ☐ ENTER SUBJECT NUMBER INTO pup.subjectNum
- ☐ RUN PROGRAM BY CLICKING AT GREEN PLAY BUTTON AT TOP OF WINDOW
- ☐ PRESS “ENTER”, AND FOCUS CAMERA ON SUBJECT WITHOUT THEM MOVING
 - IF YOU CANNOT FOCUS, HAVE SUBJECT MOVE AND TRY AGAIN
- ☐ CALIBRATE AND VALIDATE
 - Remember: C starts it; V validates; if it hangs, press space;
- ☐ REPEAT CALIBRATION A FEW TIMES UNTIL SUBJECT VALUES ARE GREEN ON HOST
- ☐ PRESS ‘O’ OR ‘ENTER’ TO PROCEED TO PROCEED IF ALL ARE GREEN
 - IF SUBJECT CANNOT CALIBRATE GREEN – TERMINATE SESSION
- ☐ PRESS SPACE TO START AND PROCEED THROUGH PRACTICE
- ☐ ENSURE SUBJECT IS GETTING ANSWERS CORRECT (all should be easy) AND UNDERSTANDS TASK
 - IF SUBJECT MISSES ON FIRST PRACTICE TRIAL – COACH

- IF SUBJECT MISSES ON SECOND AND STILL DOES NOT UNDERSTAND TASK – TERMINATE SESSION
- ☐ PRESS SPACE TO START LIGHT LEVEL TEST
- ☐ TEST WILL STOP AND CLOSE WINDOW WHEN DONE
 - IF THERE IS RED TEXT – STOP AND GET HELP
- ☐ PUPIL SCORES WILL BE ON MATLAB COMMAND WINDOW
 - Ensure there are 3 numbers for each, and that they are all different by more than 1 mm
- ☐ OFFER SUBJECT LAST OPPORTUNITY FOR QUESTIONS, HYDRATION, BATHROOM BREAK
- ☐ OPEN LumDynPupilFullV3.m
- ☐ ENTER SUBJECT ID INTO lum.subjectNum
- ☐ RUN BY PRESSING GREEN PLAY BUTTON AT TOP OF SCREEN
- ☐ PERFORM CALIBRATION PROCEDURE
 - Do not proceed unless all values are green
 - If subject cannot get green values after 3 attempts, terminate session
- ☐ USE MANILLA SLIDER TO OUTLINE EACH TRIAL
- ☐ QUICK CHECK THE TRIAL VALUES TO MAKE SURE YOU ARE ON THE RIGHT COLUMN
- ☐ CROSS OUT ANY VALUES IN THE SEQUENCE THAT THE SUBJECT MISSES
 - Do not make sounds or indicate how the subject is performing
 - Do not engage the subject in idle conversation
 - Do not make distracting noise or movements
 - Try to maintain/encourage fast trial pacing
 - If the subject is getting frustrated/discouraged encourage them to keep trying
 - Expect that subjects will make lots of mistakes, especially on the n-2 back
- ☐ WHEN THE SESSION IS COMPLETE, PROGRAM WILL CLOSE – CHECK FOR RED TEXT
- ☐ THANK SUBJECT FOR PARTICIPATION

Appendix D. NASA TLX

Figure 8.6

NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

Name	Task	Date
------	------	------

Mental Demand How mentally demanding was the task?

Very Low Very High

Physical Demand How physically demanding was the task?

Very Low Very High

Temporal Demand How hurried or rushed was the pace of the task?

Very Low Very High

Performance How successful were you in accomplishing what you were asked to do?

Perfect Failure

Effort How hard did you have to work to accomplish your level of performance?

Very Low Very High

Frustration How insecure, discouraged, irritated, stressed, and annoyed were you?

Very Low Very High

NASA TLX Sourced from National Aeronautics and Space Administration Human Systems
NASA TLX webpage on 26 January 2021 via
<https://humansystems.arc.nasa.gov/groups/TLX/index.php>.

Appendix E. CODE: designLumSetsV2.m

```
function [] = designLumSetsV2%
%
%   Creates the trail and luminance condition order for the entire
subject
%   data set for L.Temme's USAARL DVE Pupil project
%
%   Uses a specified seed for all of the randomoization so that this
code
%   can recreate the exact set if the generated files are lost or need
to
%   be created elsewhere without data-shariing
%
%   Also includes a subfunction that generates the n-back random
number
%   sequence audio files used by LumDynPupilFullV2.m
%
%   needs audiorecording of individual numbers in the same format as
%   IndNumsCA.mat in the savepath directory
%
%   V2 mixes in n-0 back into the random sequence and eliminates the
first
%   four patterns in order
%
seed = [1 2 3]; %seed for the rng so that the same set order comes out
when run again, orig set to [1,2,3]
numofSubjects = 24;% must be same as number of total trials for latin
square balanced randomization to occur
numofLuminances = 3;
numofPatterns = 12;%4 with n-1 back and 4 with n-2 back and 4 with n-
0 back (v2)
numofAudioFiles = 48;%not needed, but must be at least as many as the
number of total trials, only audio files 1-numofTotalTrials will be
assigned
numofNBackTrials = numofPatterns*numofLuminances;
numofTotalTrials = 36;
numofNumbers = 20;
numofTrialsPerLum = 12;% including all 3 n-back conditions - default
12

randNfs = 44100;%sample rate of the recorded individual numbers
timebtwnN = 1.5;%presentation timing of the nunbers in the n-back
sequence (time between start of speaking)
IndNumAudioFile = 'IndNumsCA.mat';

audiofilepath = 'C:\MATLAB\AUDIO\';
savepath = 'C:\MATLAB\ANALYSIS\'; %loaction of both below
savename = 'LumSets123V2.mat'; %mat fils saving everything
```

```

for idx1 = 1:numofLuminances:numofSubjects
    lumppreSet1 = randlatsq(numofLuminances,seed(1));
    if idx1 == 1
        lumppreSet = lumppreSet1;
    else
        lumppreSet = horzcat(lumppreSet,lumppreSet1);
    end
end

for idx2 = 1:numofPatterns:numofNBackTrials
    patpreSet1 = randlatsq(numofPatterns,seed(1));
    patpreSet2 = randlatsq(numofPatterns,seed(2));
    patpreSet3 = randlatsq(numofPatterns,seed(3));
    if idx2 == 1
        patpreSetA = patpreSet1;
        patpreSetB = patpreSet2;
        patpreSetC = patpreSet3;
    else
        patpreSetA = horzcat(patpreSetA,patpreSet1);
        patpreSetB = horzcat(patpreSetB,patpreSet2);
        patpreSetC = horzcat(patpreSetC,patpreSet3);
    end
end

audiopreSet = randlatsq(numofTotalTrials,seed(1));

for subj = 1:numofSubjects
    lumOrder(subj,:) = lumppreSet(:,subj); %1 is dimmest, 3 is
brightest
    patOrderA(subj,:) = patpreSetA(:,subj); %1-3 is n-1 back, 2-6 is
n-2 back
    patOrderB(subj,:) = patpreSetB(:,subj);
    patOrderC(subj,:) = patpreSetC(:,subj);
    audOrder(subj,:) = audiopreSet(:,subj); %directly refs audio
recordings 1-36
end

rng(seed(1));
nbackNumSets = randi(9,numofNumbers,numofAudioFiles);

csvwrite('nbackNumSets123.csv',nbackNumSets);
nbackPracSet1 = [1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9 1 2];
nbackPracSet2 = [9 8 7 6 5 4 3 2 1 9 8 7 6 5 4 3 2 1 9 8];
%generates n-back audio files from individual num recordings based on
params above
nbackAudioMaker(audiofilepath,IndNumAudioFile,randNfs,timebtwN,nbackNu
mSets,nbackPracSet1,nbackPracSet2);

for subj = 1:numofSubjects
    lumSet1(1:numofTrialsPerLum) = lumOrder(subj,1);
    lumSet1(numofTrialsPerLum+1:numofTrialsPerLum*2) =

```

```

lumOrder(subj,2);
    lumSet1(numofTrialsPerLum*2+1:numofTrialsPerLum*3) =
lumOrder(subj,3);
    lumSet(subj,:) = lumSet1;
    %luminance change pattern and n-back condition order

    patSet(subj,:) =
horzcat(patOrderA(subj,:),patOrderB(subj,:),patOrderC(subj,:));
%master order set with a single value for each trial

    % extract the perscribed order for the number reading audio files
audSet(subj,:) = audOrder(subj,:);

    for idx1 = 1:length(patSet)
        if patSet(subj,idx1)<=4
            nbackSet(subj,idx1) = 0;
        end
        if patSet(subj,idx1)>4 && patSet(subj,idx1)<= 8
            patSet(subj,idx1) = patSet(subj,idx1)-4;
            nbackSet(subj,idx1) = 1;
        end
        if patSet(subj,idx1) > 8
            patSet(subj,idx1) = patSet(subj,idx1)-8;
            nbackSet(subj,idx1) = 2;
        end
    end
end

end

%saves the variables created to savename (does not include the full
length
%audio files for the n-back -- nbackAudioMaker saves those
cd(savepath);
save(savename,'lumSet','patSet','nbackSet','audSet','nbackNumSets');

end

%% EMBEDDED FUNTIONS FOR ABOVE OPERATIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [] =
nbackAudioMaker(audiofilepath,IndNumAudioFile,randNfs,timebtwN,nbackNu
mSets,nbackPracSet1,nbackPracSet2)
%generates the full length N-Back number reading audio files per the
order
%perscribed in the parent function
% needs IndNumsCA (recording of individual numbers) to be in the
savepath
% directory

```

```

%%
cd(audiofilepath);
load(IndNumAudioFile,'nums');

sampbtwN = randNfs*timebtwN-length(nums);
filler = zeros(1,sampbtwN);

for set = 1:size(nbackNumSets,2)
    randNum = 0;
    for num = 1:size(nbackNumSets,1)
        randNum =
horzcat(randNum,nums(nbackNumSets(num,set),:),filler);
    end
    nbackNumVals(1,:) = nbackNumSets(:,set);
    savename = sprintf('%s%d%s','RandNumsSoundSet',set,'.mat');
    save(savename,'randNum','randNfs','nbackNumVals');
end
pracNum1 = 0;
pracNum2 = 0;
for num1 = 1:length(nbackPracSet1)
    pracNum1 = horzcat(pracNum1,nums(nbackPracSet1(num1),:),filler);
    pracNum2 = horzcat(pracNum2,nums(nbackPracSet2(num1),:),filler);
end

nbackPracVals(1,:) = nbackPracSet1;
nbackPracVals(2,:) = nbackPracSet2;
savename = sprintf('%s%s','PracNumsSoundSet1.mat');
save(savename,'pracNum1','randNfs','nbackPracVals');
savename = sprintf('%s%s','PracNumsSoundSet2.mat');
save(savename,'pracNum2','randNfs','nbackPracVals');

end

function [M] = randlatsq(N,seed)
%
%   uses latsq and creates and nxn matrix = M, but starts from a
random order as dictated by seed
%   taken from mathworks file exchange 05/29/19 by CA
%   - posted by user: Jos (no other author info available)
%%
rng(seed);
V = randperm(N);
M = V(latsq(numel(V)));
end

function [M] = latsq(N)
%
%   creates a NxN latin square: it generates N by N matrix in which
%   each value only appears once in each row and column.
%
%   taken from mathworks file exchange 05/29/19 by CA

```

```
% - posted by user: Jos (no other author info available)
%%

M = [1:N ; ones(N-1,N)];
M = rem(cumsum(M)-1,N)+1;
end
```

Appendix F. CODE: LightLevelTest.m

```
%LightLevelTest.m
%
%   This should be run daily before collecting DVE-Pupil Data
%
%   This program just runs through the 3 luminance levels for
%   the study so that measurements can be taken with the
%   luminance meter to be sure everything is working correctly
%
%   Be sure to put today date in below in the same format
%   written by CA for L.Temme & USAARL DVE Pupil Study
%   last modified by CA on 6.5.2019
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

chk.date = '071819'; % use the format MMDDYY

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DO NOT CHANGE ANY VALUES BELOW THIS LINE
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

chk.lumRGBvalues = [40 60 200];%rgb values of the luminance levels -
default [40 60 200]

chk.hedge = [0.001 0.999]; %8" aperature = 0.315 and 0.675 of screen
to left and right edge of aperture (~5 mm overlap) [0.001 0.999]

chk.vedge = [0.001 0.999]; %8" aperature= 0.18  and 0.815 to top and
bottom of screen edge [0.001 0.999]

chk.dimwid = [0.006 0.006]; %%screen width to shrink aperture window
to have surround peek for dimmest condition [l/r u/d]

messageText = sprintf('%s','Pupil Calib Complete, Press the spacebar
to start Luminance tests and advance to the next screen');
pupilcalibText = sprintf('%s','Place the pupil card in front of camera
and press space to start calibration');

savename = sprintf('%s%s','LumTest',chk.date);
chk.edfFile = sprintf('%s%s','p',chk.date);

savepath = 'C:\MATLAB\DATA\';

screens = Screen('Screens');
screenNumber = max(screens); %if in multidisplay mode, specify which
screen to show the stim on - default 1

luminance = chk.lumRGBvalues./255;
%%
PsychDefaultSetup(2);
Screen('Preference', 'VisualDebugLevel', 1);
```

```

% Stop the program from quitting if the sync tests fail in testing
mode
% BE SURE TO REMOVE THIS AND TEST BEFORE PROPER EXPERIMENTS
% if pup.subjectNum == 0
%     Screen('Preference', 'SkipSyncTests', 1)
% end
Screen('Preference', 'DefaultFontSize', 24);
%redirect the typing location to the command window so that
calibration
%keypresses are not entered as changes in the code of this script
commandwindow;

% Open a graphics window on the main screen
% using the PsychToolbox's PsychImaging function.
[window, windowRect]=PsychImaging('OpenWindow', screenNumber,
0,[],32,2);
Screen(window,'BlendFunction',GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
[width, height]=Screen('WindowSize', screenNumber);
[xCenter, yCenter] = RectCenter(windowRect);

%%
el=EyelinkInitDefaults(window);

% We are changing calibration to a black background with white
targets,
% no sound and smaller targets
el.backgroundcolour = BlackIndex(el.window);
el.msgfontcolour    = WhiteIndex(el.window);
el.imgtitlecolour   = WhiteIndex(el.window);
el.targetbeep = 0;
el.calibrationtargetcolour = WhiteIndex(el.window);
el.calibrationtargetsize= .5;
el.calibrationtargetwidth=0.25;

% make changes to the calibration structure to take effect
EyelinkUpdateDefaults(el);

%% Initialization of the connection with the Eyelink Gazetracker.
% EyelinkInit(0);
Eyelink('Initialize');

i = Eyelink('Openfile', chk.edfFile);
%% SET UP TRACKER CONFIGURATION
% Setting the proper recording resolution, proper calibration type,
% as well as the data file content;
Eyelink('command', 'add_file_preamble_text 'Recorded by
LightLevelTest.m'');
% This command is crucial to map the gaze positions from the tracker
to
% screen pixel positions to determine fixation
Eyelink('command','screen_pixel_coords = %ld %ld %ld %ld', 0, 0,
width-1, height-1);

```

```

WaitSecs(0.1); %need to give the host pc time to register between
successive commands
Eyelink('message', 'DISPLAY_COORDS %ld %ld %ld %ld', 0, 0, width-1,
height-1);
WaitSecs(0.1);
% set calibration type.
Eyelink('command', 'calibration_type = HV5');
WaitSecs(0.1);
% you must send this command with value NO for custom calibration
% you must also reset it to YES for subsequent experiments
Eyelink('command', 'generate_default_targets = NO');
WaitSecs(0.1);
% STEP 5.1 modify calibration and validation target locations
Eyelink('command', 'calibration_samples = 6');
WaitSecs(0.1);
Eyelink('command', 'calibration_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command', 'calibration_targets = %d,%d %d,%d %d,%d %d,%d
%d,%d',...
    round(width/2),round(height/2),
    round(width/2),round(height*0.25), round(width/2),round(height -
height*0.25), round(width*0.25),round(height/2), round(width -
width*0.25),round(height/2));
WaitSecs(0.1);
Eyelink('command', 'validation_samples = 6');
WaitSecs(0.1);
Eyelink('command', 'validation_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command', 'validation_targets = %d,%d %d,%d %d,%d %d,%d
%d,%d',...
    round(width/2),round(height/2),
    round(width/2),round(height*0.25), round(width/2),round(height -
height*0.25), round(width*0.25),round(height/2), round(width -
width*0.25),round(height/2) );
WaitSecs(0.1);
% % set parser (conservative saccade thresholds)
% Eyelink('command', 'saccade_velocity_threshold = 35');
% WaitSecs(0.1);
% Eyelink('command', 'saccade_acceleration_threshold = 9500');
% WaitSecs(0.1);
% set EDF file contents - the data that will be recorded by the host
PC -
% DO NOT CHANGE!!
% remote mode possible add HTARGET ( head target)
Eyelink('command', 'file_event_filter =
LEFT,RIGHT, FIXATION, SACCAD, BLINK, MESSAGE, BUTTON, INPUT');
WaitSecs(0.1);
Eyelink('command', 'file_sample_data =
LEFT,RIGHT,GAZE,HREF,AREA,GAZERES,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% set link data - the values that we can query during the recording
Eyelink('command', 'link_event_filter =

```

```

LEFT,RIGHT, FIXATION, SACCAD, BLINK, MESSAGE, BUTTON, FIXUPDATE, INPUT');
WaitSecs(0.1);
Eyelink('command', 'link_sample_data =
LEFT,RIGHT,GAZE,GAZERES,AREA,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% calibration/drift correction target - controller not used but
removing
% this makes accepting fixation on the display pc KB no longer work
Eyelink('command', 'button_function 5 "accept_target_fixation"');
WaitSecs(0.1);
Eyelink('Command', 'set_idle_mode'); % seems to be a good practice to
ensure that the recording starts quickly
WaitSecs(0.5);

Screen('FillRect', window, [0 0 0],...
    [round(windowRect(3)*chk.hedge(1))
round(windowRect(4)*chk.vedge(1))...
    round(windowRect(3)*chk.hedge(2))
round(windowRect(4)*chk.vedge(2))]);

Screen('DrawText', window, pupilcalibText, 200, yCenter, [1 1 1]);

Screen('Flip', window);
KbStrokeWait;

Eyelink('StartRecording');
WaitSecs(0.5); %there is a delay between this command is issues and
when eye data starts being collected - good to give it a few hundred
ms
Eyelink('Message', 'PUPIL CARD CALIB START');
WaitSecs(10);
Eyelink('StopRecording');
WaitSecs(0.1);
Eyelink('Command', 'set_idle_mode'); % seems to be a good practice to
ensure that the recording starts quickly
WaitSecs(0.1);
Eyelink('CloseFile');
WaitSecs(0.5);
status=Eyelink('ReceiveFile');

WaitSecs(0.5);
Eyelink('Shutdown');

%% START LUMINANCE SCREEN CALIBRATIONS
% START SPLASH INFO
Screen('FillRect', window, [0 0 0],...
    [round(windowRect(3)*chk.hedge(1))
round(windowRect(4)*chk.vedge(1))...
    round(windowRect(3)*chk.hedge(2))
round(windowRect(4)*chk.vedge(2))]);

Screen('DrawText', window, messageText, 200, yCenter, [1 1 1]);

```

```

Screen('Flip', window);
KbStrokeWait;

% BLACK SCREEN
Screen('FillRect', window, [0 0 0],...
    [round(windowRect(3)*chk.hedge(1))
round(windowRect(4)*chk.vedge(1))...
    round(windowRect(3)*chk.hedge(2))
round(windowRect(4)*chk.vedge(2))]);

Screen('Flip', window);
KbStrokeWait;

% DIM LUMINANCE LEVEL SCREEN
Screen('FillRect', window, [luminance(1) luminance(1) luminance(1)]);
Screen('FillRect', window, [0 0 0],...
    [round(windowRect(3)*(chk.hedge(1)+chk.dimwid(1)))
round(windowRect(4)*(chk.vedge(1)+chk.dimwid(2)))...
    round(windowRect(3)*(chk.hedge(2)-chk.dimwid(1)))
round(windowRect(4)*(chk.vedge(2)-chk.dimwid(2)))]);

Screen('Flip', window);
KbStrokeWait;

% MIDDLE LUMINANCE LEVEL SCREEN
Screen('FillRect', window, [luminance(2) luminance(2)
luminance(2)],...
    [round(windowRect(3)*chk.hedge(1))
round(windowRect(4)*chk.vedge(1))...
    round(windowRect(3)*chk.hedge(2))
round(windowRect(4)*chk.vedge(2))]);

Screen('Flip', window);
KbStrokeWait;

% BRIGHT LUMINANCE LEVEL SCREEN
Screen('FillRect', window, [luminance(3) luminance(3)
luminance(3)],...
    [round(windowRect(3)*chk.hedge(1))
round(windowRect(4)*chk.vedge(1))...
    round(windowRect(3)*chk.hedge(2))
round(windowRect(4)*chk.vedge(2))]);

Screen('Flip', window);
KbStrokeWait;

Screen('CloseAll');

% PROMPT FOR THE METER'S DATA VALUES

chk.vals(1) = inputdlg('BLACK SCREEN LUMINANCE?');

```

```
chk.vals(2)= inputdlg('DIM SCREEN LUMINANCE?');  
chk.vals(3)= inputdlg('MIDDLE SCREEN LUMINANCE?');  
chk.vals(4)= inputdlg('BRIGHT SCREEN LUMINANCE?');  
  
cd(savepath);  
save(savename,'chk');  
  
clear all;
```

Appendix G. CODE: LumDynPupilPractV1.m

```
%LumDynPupilPractV1.m
%
% Tests EyeLink connectivity and recording ability.
% Runs the pre-experiment practice for Calibration and nback
Tasks.
% Then give a preview of the luminance levels and collects
subject
% pupil size for all three luminance tiers.
%
% THESE DATA ARE RECORDED - ENSURE SUBJECT NUMBER IS ACCURATE!!
%
% Saves all eye data collected to a data file
S<subjectNum>PRE.edf
% Saves all taks information and pupil size to
S<subjectNum>PrePups.mat
%
% V1 written by CJA on 07012019 for USAARL DVE Pupil Study
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
pup.subjectNum = 0;

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pup.hedge = [0.001 0.999]; % sets the width of the screen edge -
in case edge needs to be shrunk for aperture - no aperture =
[.001 .999]

pup.vedge = [0.001 0.999]; % sets the height boundaries of the
screen edge - in case edge needs to be shrunk for aperture - no
aperture = [.001 .999]

pup.dimwid = [0.006 0.006]; %%screen width to shrink aperture
window to have surround peek for dimmest condition [l/r u/d]

pup.numPsamples = 40; % number of pupil diameter samples to
collect and average to get pupil diameter

pup.pupSampWaitT = 25; %# of secs to wait before collecting
pupil size samples

pup.textleft = 750; %left justification of the text instruction
in pixels
```

```

pup.texthigh = [340 380 420 530 640]; %height tiers of text
instructions in pizels

pup.textfont = 24; % font of text in pixels

pup.numPracSecs = 30;% length of the practice counting trials

pup.numPreSecs = 1; % minimum length of time to present the
instruction screens before listening for a keypress - default 5
secs

pup.lumTestSeq = [40 60 200]; %RGB luminance values used for
study ** MAKE SURE THESE MATCH LumDynPupilFull

fixCrossDimPix = 10; %length in pixels of the arms of the
fixation cross - default 20

lineWidthPix = 2; % width in pixels of the arms of the fixation
cross - default 4

numofprac = 10;

pup.soundpath = 'C:\MATLAB\AUDIO\'; %where to find the audio
files
pup.edfFilePath = 'C:\MATLAB\DATA\'; %where to save the data
files
%% test pupil calib info for converting pupil size to cam
pixels.
% pup.testPupilAreainPix = [2050 1943]; %in cam pix (output
evt.pa) - set for dextop head-fixed mode
pup.testPupilAreainPix = [350 423]; %set for destop remote mode
with 16mm lens
pup.testPupilSizeinMM = [5 5];%in mm, measured from test pupil
card

%% Do calcs and conversions for stim and measure setup
pup.lumTestSeqP = pup.lumTestSeq./255;

%calculate the conversion constant for pix to mm conversion.
testPDpix = (sqrt(pup.testPupilAreainPix)./pi).*2;
pup.pupilLCF = testPDpix(1)/pup.testPupilSizeinMM(1);
pup.pupilRCF = testPDpix(2)/pup.testPupilSizeinMM(2);

%%

```

```

pup.edfFile = sprintf('%s%d%s', 'S', pup.subjectNum, 'PRE');

pup.edfFullName =
sprintf('%s%s%s', pup.edfFilePath, pup.edfFile, '.edf');

%%
screens = Screen('Screens');
screenNumber = max(screens); %if in multidisplay mode, specify
which screen to show the stim on - default 1

%%
PsychDefaultSetup(2);
Screen('Preference', 'VisualDebugLevel', 1);
% Stop the program from quitting if the sync tests fail in
testing mode
% BE SURE TO REMOVE THIS AND TEST BEFORE PROPER EXPERIMENTS
% if pup.subjectNum == 0
%     Screen('Preference', 'SkipSyncTests', 1)
% end
Screen('Preference', 'DefaultFontSize', pup.textfont);
%redirect the typing location to the command window so that
calibration
%keypresses are not entered as changes in the code of this
script
commandwindow;

% Open a graphics window on the main screen
% using the PsychToolbox's PsychImaging function.
[window, windowRect]=PsychImaging('OpenWindow', screenNumber,
0, [], 32, 2);
Screen(window, 'BlendFunction', GL_SRC_ALPHA,
GL_ONE_MINUS_SRC_ALPHA);
[width, height]=Screen('WindowSize', screenNumber);
[xCenter, yCenter] = RectCenter(windowRect);

% Coordinates for the fixation cross
xCoords = [-fixCrossDimPix fixCrossDimPix 0 0];
yCoords = [0 0 -fixCrossDimPix fixCrossDimPix];
allCoords = [xCoords; yCoords];

% Provide Eyelink with details about the graphics environment
% and perform some initializations.
el=EyelinkInitDefaults(window);

% We are changing calibration to a black background with white
targets,

```

```

% no sound and smaller targets
el.backgroundcolour = BlackIndex(el.window);
el.msgfontcolour = WhiteIndex(el.window);
el.imgtitlecolour = WhiteIndex(el.window);
el.targetbeep = 0;
el.calibrationtargetcolour = WhiteIndex(el.window);
el.calibrationtargetsize= .5;
el.calibrationtargetwidth=0.25;

% make changes to the calibration structure to take effect
EyelinkUpdateDefaults(el);

%% Initialization of the connection with the Eyelink
Gazetracker.

EyelinkInit(0);

% Navigate to the folder containing data files
cd(pup.edfFilePath);

% Check to make sure that a file with this subjectNum has not
previously
% been collected (ensure previously collected data is not
overwritten)
if pup.subjectNum~=0 && isfile(pup.edfFullName) == 0 %subjectnum
not set to test and an edf file for this subject does not exist
    i = Eyelink('Openfile', pup.edfFile);
    if i~=0 %0 means that the procedure worked - any other
number means a fail
        fprintf('Cannot create EDF file for ''%s'' ',
pup.edfFile);
        sca;
        Eyelink('Shutdown');
        return;
    end
elseif pup.subjectNum == 0 %test mode designated, will overwrite
last edf and *.mat files with SUB0 designation
    i = Eyelink('Openfile', pup.edfFile);
    if i~=0
        fprintf('Cannot create EDF file for ''%s'' ',
pup.edfFile);
        sca;
        Eyelink('Shutdown');
        return;
    end
else
    fprintf('EDF file already exists for this subject!!\n Check

```

```

the SubjectNum variable for accuracy! '%s' ', pup.edfFile);
    sca;
    Eyelink('Shutdown');
    return;
end

%% SET UP TRACKER CONFIGURATION
% Setting the proper recording resolution, proper calibration
type,
% as well as the data file content;
Eyelink('command', 'add_file_preamble_text 'Recorded by
LumDynPupilFullV2 by CA'');
% This command is crucial to map the gaze positions from the
tracker to
% screen pixel positions to determine fixation
Eyelink('command', 'screen_pixel_coords = %ld %ld %ld %ld', 0, 0,
width-1, height-1);
WaitSecs(0.1); %need to give the host pc time to register
between successive commands
Eyelink('message', 'DISPLAY_COORDS %ld %ld %ld %ld', 0, 0,
width-1, height-1);
WaitSecs(0.1);
% set calibration type.
Eyelink('command', 'calibration_type = HV5');
WaitSecs(0.1);
% you must send this command with value NO for custom
calibration
% you must also reset it to YES for subsequent experiments
Eyelink('command', 'generate_default_targets = NO');
WaitSecs(0.1);
% STEP 5.1 modify calibration and validation target locations
Eyelink('command', 'calibration_samples = 6');
WaitSecs(0.1);
Eyelink('command', 'calibration_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command', 'calibration_targets = %d,%d %d,%d %d,%d %d,%d
%d,%d', ...
    round(width/2), round(height/2),
    round(width/2), round(height*0.25), round(width/2), round(height
- height*0.25), round(width*0.25), round(height/2), round(width
- width*0.25), round(height/2));
WaitSecs(0.1);
Eyelink('command', 'validation_samples = 6');
WaitSecs(0.1);
Eyelink('command', 'validation_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command', 'validation_targets = %d,%d %d,%d %d,%d %d,%d

```

```

%d,%d',...
    round(width/2),round(height/2),
round(width/2),round(height*0.25), round(width/2),round(height
- height*0.25), round(width*0.25),round(height/2), round(width
- width*0.25),round(height/2) );
WaitSecs(0.1);
% % set parser (conservative saccade thresholds)
% Eyelink('command', 'saccade_velocity_threshold = 35');
% WaitSecs(0.1);
% Eyelink('command', 'saccade_acceleration_threshold = 9500');
% WaitSecs(0.1);
% set EDF file contents - the data that will be recorded by the
host PC -
% DO NOT CHANGE!!
% remote mode possible add HTARGET ( head target)
Eyelink('command', 'file_event_filter =
LEFT,RIGHT,FIXATION,SACCADE,BLINK,MESSAGE,BUTTON,INPUT');
WaitSecs(0.1);
Eyelink('command', 'file_sample_data =
LEFT,RIGHT,GAZE,HREF,AREA,GAZERES,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% set link data - the values that we can query during the
recording
Eyelink('command', 'link_event_filter =
LEFT,RIGHT,FIXATION,SACCADE,BLINK,MESSAGE,BUTTON,FIXUPDATE,INPUT
');
WaitSecs(0.1);
Eyelink('command', 'link_sample_data =
LEFT,RIGHT,GAZE,GAZERES,AREA,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% calibration/drift correction target - controller not used but
removing
% this makes accepting fixation on the display pc KB no longer
work
Eyelink('command', 'button_function 5
"accept_target_fixation"');
WaitSecs(0.1);

% Hide the mouse cursor if testmode (sub0) not engaged - major
pain in the ass if
% the program fails - so only leave on when are certain code it
is working
if pup.subjectNum ~= 0
    Screen('HideCursorHelper', window);
end

% enter Eyetracker camera setup mode, calibration and validation

```

```

EyelinkDoTrackerSetup(el);
WaitSecs(1);
Eyelink('Command', 'set_idle_mode'); % seems to be a good
practice to ensure that the recording starts quickly
WaitSecs(0.5);
Eyelink('StartRecording');
WaitSecs(0.5); %there is a delay between this command is issues
and when eye data starts being collected - good to give it a few
hundred ms
keypressText = sprintf('%s','ARE YOU READY TO START?');
practiceInst1 = sprintf('%s','READ THE TASK INSTRUCTIONS NOW');
practiceInst2 = sprintf('%s','WE WILL BEGIN THE PRACTICE SET');
practiceInst3 = sprintf('%s','WHEN YOU ARE READY TO BEGIN. ');
practiceInst4 = sprintf('%s','PRACTICE SET COMPLETE!');

practiceN0 = sprintf('%s','N-0 BACK PRACTICE');
practiceN1 = sprintf('%s','N-1 BACK PRACTICE');
practiceN2 = sprintf('%s','N-2 BACK PRACTICE');
practiceN3 = sprintf('%s','N-3 BACK PRACTICE');

WaitSecs(0.1)
Eyelink('Message','PRACTICE SESSION START');
SoundSet =
sprintf('%s%s%d%s',pup.soundpath,'PracNumsSoundSet.mat');
load(SoundSet,'pracNum','randNfs');

Screen('TextSize', window, pup.textfont);

Screen('FillRect', window, [.2 .2 .2],...
    [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
    round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
% make the instructions appropriate for the task as is
% set by the subjectNum determined trial order
Screen('DrawText', window, practiceInst1, pup.textleft,
pup.texthigh(1));
Screen('DrawText', window, practiceInst2, pup.textleft,
pup.texthigh(2));
Screen('DrawText', window, practiceInst3, pup.textleft,
pup.texthigh(3));
Screen('DrawText', window, keypressText, pup.textleft,
pup.texthigh(5));
%shows what we just make and marks the edf file
Screen('Flip', window);
WaitSecs(pup.numPreSecs);
%wait on first frame for keypress from experimenter

```

```

KbStrokeWait;

for prac = 1:numofprac

    Screen('FillRect', window, [.2 .2 .2],...
        [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
        round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
    if prac == 1
        Screen('DrawText', window, practiceN0, pup.textleft,
pup.texthigh(4));
        %shows what we just make and marks the edf file
    elseif prac > 1 && prac <=4
        Screen('DrawText', window, practiceN1, pup.textleft,
pup.texthigh(4));
    elseif prac > 4 && prac <=7
        Screen('DrawText', window, practiceN2, pup.textleft,
pup.texthigh(4));
    elseif prac > 7
        Screen('DrawText', window, practiceN3, pup.textleft,
pup.texthigh(4));
    end
    Screen('Flip', window);

    if prac == 4
        SoundSet =
sprintf('%s%s%d%s',pup.soundpath,'RandNumsSoundSet46.mat');
        load(SoundSet,'randNum','randNfs');
        pracNum = randNum;
    elseif prac == 7
        SoundSet =
sprintf('%s%s%d%s',pup.soundpath,'RandNumsSoundSet47.mat');
        load(SoundSet,'randNum','randNfs');
        pracNum = randNum;
    elseif prac == 10
        SoundSet =
sprintf('%s%s%d%s',pup.soundpath,'RandNumsSoundSet48.mat');
        load(SoundSet,'randNum','randNfs');
        pracNum = randNum;
    else
        SoundSet =
sprintf('%s%s%d%s',pup.soundpath,'PracNumsSoundSet.mat');
        load(SoundSet,'pracNum','randNfs');
    end

    WaitSecs (pup.numPreSecs);

```

```

    %wait on first frame for keypress from experimenter
    KbStrokeWait;

    Screen('FillRect', window, [.2 .2 .2],...
        [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
        round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
    Screen('DrawLines', window, allCoords,...
        lineWidthPix, [1 0 0], [xCenter yCenter], 2);
    Screen('Flip', window);
    sound(pracNum,randNfs);
    WaitSecs(pup.numPracSecs);
    clear sound;

end

Screen('FillRect', window, [.2 .2 .2],...
    [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
    round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
Screen('DrawText', window, practiceInst4, pup.textleft,
pup.texthigh(4));
Screen('Flip', window);
WaitSecs(pup.numPreSecs);
KbStrokeWait;

PupilTestInst1 = sprintf('%s','NOW WE WILL TEST YOUR PUPIL');
PupilTestInst2 = sprintf('%s','RESPONSES TO THE LIGHT LEVELS');
PupilTestInst3 = sprintf('%s','PLEASE LOOK AT THE CROSS');
PupilTestInst4 = sprintf('%s','WHILE THE SCREEN CHANGES');

Screen('FillRect', window, [.2 .2 .2],...
    [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
    round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
% make the instructions appropriate for the task as is
% set by the subjectNum determined trial order
Screen('DrawText', window, PupilTestInst1, pup.textleft,
pup.texthigh(1));
Screen('DrawText', window, PupilTestInst2, pup.textleft,
pup.texthigh(2));
Screen('DrawText', window, PupilTestInst3, pup.textleft,
pup.texthigh(3));

```

```

Screen('DrawText', window, PupilTestInst4, pup.textleft,
pup.texthigh(4));
%shows what we just make and marks the edf file
Screen('Flip', window);
WaitSecs(pup.numPreSecs);
%wait on first frame for keypress from experimenter
KbStrokeWait;

for x = 1:length(pup.lumTestSeqP)

    %           Eyelink('Message', 'TRIALID %d - LUMIN
#d',x,LumTestSeq(x));
    % This supplies the title at the bottom of the eyetracker
display
    if x == 1
        Screen('FillRect', window, [pup.lumTestSeqP(x)
pup.lumTestSeqP(x) pup.lumTestSeqP(x)]);
        Screen('FillRect', window, [0 0 0],...
            [round(windowRect(3)*(pup.hedge(1)+pup.dimwid(1)))
round(windowRect(4)*(pup.vedge(1)+pup.dimwid(2)))...
            round(windowRect(3)*(pup.hedge(2)-pup.dimwid(1)))
round(windowRect(4)*(pup.vedge(2)-pup.dimwid(2)))]);
    else
        Screen('FillRect', window, [pup.lumTestSeqP(x)
pup.lumTestSeqP(x) pup.lumTestSeqP(x)],...
            [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
            round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
    end
    Screen('DrawLines', window, allCoords,...
        lineWidthPix, [0 .5 0], [xCenter yCenter], 2);

    % Flip to the screen
    Screen('Flip', window);
    Eyelink('Message', 'STIM ON');
    WaitSecs(pup.pupSampWaitT);
    %draw cross to indicate critical fixation period
    if x == 1
        Screen('FillRect', window, [pup.lumTestSeqP(x)
pup.lumTestSeqP(x) pup.lumTestSeqP(x)]);
        Screen('FillRect', window, [0 0 0],...
            [round(windowRect(3)*(pup.hedge(1)+pup.dimwid(1)))
round(windowRect(4)*(pup.vedge(1)+pup.dimwid(2)))...
            round(windowRect(3)*(pup.hedge(2)-pup.dimwid(1)))
round(windowRect(4)*(pup.vedge(2)-pup.dimwid(2)))]);
    else

```

```

        Screen('FillRect', window, [pup.lumTestSeqP(x)
pup.lumTestSeqP(x) pup.lumTestSeqP(x)],...
        [round(windowRect(3)*pup.hedge(1))
round(windowRect(4)*pup.vedge(1))...
        round(windowRect(3)*pup.hedge(2))
round(windowRect(4)*pup.vedge(2))]);
    end
    Screen('DrawLines', window, allCoords,...
        lineWidthPix, [.5 0 0], [xCenter yCenter], 2);
    Screen('Flip', window);
    WaitSecs(2);
    Eyelink('Message', 'SAMPLE PUPIL');
    for idx = 1:pup.numPsamples

        evt{idx} = Eyelink( 'NewestFloatSample');
        pup.pupilA(idx,:) = evt{idx}.pa;
        pup.pupilDsmpl(idx,:) =
((sqrt(pup.pupilA(idx,:))./pi).*2);
        pup.pupilDsmplL(idx) = pup.pupilDsmpl(idx,1)/pup.pupilLCF;
        pup.pupilDsmplR(idx) = pup.pupilDsmpl(idx,2)/pup.pupilRCF;
        WaitSecs(0.05);

    end
    pup.pupilDLfull(x) = mean(pup.pupilDsmplL);
    pup.pupilDRfull(x) = mean(pup.pupilDsmplR);

    Eyelink('Message', 'STIM OFF');

end

%black out screen to indicate session end
Screen('FillRect', window, [0 0 0]);

Screen('Flip', window);
Eyelink('Message', 'SESSION END');
% KbStrokeWait; %keypress to exit screen
WaitSecs(0.1);

Eyelink('StopRecording');
WaitSecs(0.5);
Eyelink('Command', 'set_idle_mode');
WaitSecs(0.5);
Eyelink('CloseFile');
WaitSecs(0.5);

```

```

% download data file
cd(pup.edfFilePath);
edfFilename = sprintf('%sPups.mat',pup.edfFile);
save(edfFilename,'evt','pup','el');

try
    fprintf('Receiving data file ''%s''\n', pup.edfFile );
    status=Eyelink('ReceiveFile');
    if status > 0
        fprintf('ReceiveFile status %d\n', status);
    end
    if 2==exist(pup.edfFile, 'file')
        fprintf('Data file ''%s'' can be found in ''%s''\n',
pup.edfFile, pwd );
    end
catch %#ok<*CTCH>
    fprintf('Problem receiving data file ''%s''\n', pup.edfFile
);
end
WaitSecs(0.1);

Eyelink('Shutdown');
Screen('CloseAll');

pup.pupilDLfull
pup.pupilDRfull

```

Appendix H. CODE: LumDynPupilFullV3.m

```
function [lum,el] = LumDynPupilFullV3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LumDynPupilFullV3.m
% [lum,el] = LumDynPupilFullV3
%
% Displays a set of luminances and luminance change patterns,
% by subject, as set by the parameters below, in an order
% perscribed by the mat file LumSets123.mat, which is generated
% by designLumSets.m
%
% V3 changes the dimmest condition to change luminance at the
% outside and
% black inside to cut light instead of using an filter over the
% aperture
%
% lum = all of the test and display paramters
% el = all of the eyelink parameters
% both are saved in a file SUB<lum.subjectNum>.mat
%
% Requires that the display be connected to a functional eyelink
% host-PC as mandated by SR-Research for proper operation
% Upon successful completion, will copy created edf from the
% host-PC to the edfFilePath specified below.
% *if this fails, no data is lost - file must be manually
% tranferred from the host-PC /etc/data/ directory
%
% Written in MATLAB R2018b
% requires Psychtoolbox3, GStreamer, and the Eyelink Developers
% Kit.
%
% Requires that the lum.subjectNum is specified as a number
% -will not run if that subject num has already been used
% or is outside of the range specified in LumSets123.mat
% (currently 24 subjects long)
%
% ** use subjectNum = 0 for testing purposes as it will not
% error out due to files already being present.
%
% written by CA for L.Temme & USAARL DVE Pupil Study
% last modified by CA on 6.5.2019
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

%% SUBJECT INFORMATION - BE SURE THIS IS SET, ACCURATE, AND
SAVED BEFORE STARTING EXPERIMENT BY CLICKING THE "Run" BUTTON
ABOVE

lum.subjectNum = 0; % Subject ID number, use 0 for testing

%%
%%
%%

%% DEBUGGING ONLY DO NOT CHANGE FOR ACTUAL STUDIES

lum.testSet = 2; %which randomization set to use for testing,
lum.subjectNum must be set to 0!

%% save, order, and sound path locations

soundpath = 'C:\MATLAB\AUDIO\';

edfFilePath = 'C:\MATLAB\DATA\';

lum.orderFile = 'C:\MATLAB\ANALYSIS\LumSets123V2.mat'; %V2 to
use full randomization of all 3 nback tasks

lum.edfFile = sprintf('%s%d', 'SUB', lum.subjectNum);

if lum.subjectNum == 0

lum.edfFile = sprintf('%s%d', 'SUB0T', lum.testSet);

end

edfFullName = sprintf('%s%s%s', edfFilePath, lum.edfFile, '.edf');

%% Parameters for the luminance trials - be sure these are
accurate and not changed between subjects

lum.numSecs = 5; %length of presentation in secs - default 30
secs

lum.numPreSecs = 1; % minimum length of time to present the
instruction screens before listening for a keypress - default 5
secs

lum.numPracSecs = 30; % length of the practice counting trials

lum.LumChgPct = 0.25; % percent of the luminance to change from
mean value in either direction - default 25% (0.25)

lum.baselinePupMin = .5; %duration of the baseline/adaptation
period in minutes - default 4 minutes

lum.baselinePupT = lum.baselinePupMin*60; %duration of the
baseline luminance period in secs

lum.lumChgFreqs = [1 0.2 0.1]; %freqs of change in hz from
fastest to slowest - default [1 0.2 0.1]

```

lum.lumRGBvalues = [40 60 200]; %rgb values of the luminance
levels - default [40 40 200]

lum.hedge = [0.001 0.999]; %8" aperature = 0.315 and 0.675 of
screen to left and right edge of aperture (~5 mm overlap)

lum.vedge = [0.001 0.999]; %8" aperature= 0.18 and 0.815 to top
and bottom of screen edge

lum.dimwid = [0.006 0.006]; %%screen width to shrink aperture
window to have surround peek for dimmest condition [l/r u/d]

lum.textleft = 750;

lum.textleft1 = 400;

lum.texthigh = [340 380 420 530 640 300];

lum.textfont = 24;

%Checks the screens attached for determining which to display on
%if cloning use this
screens = Screen('Screens');
screenNumber = max(screens); %if in multidisplay mode, specify
which screen to show the stim on - default 1

%is using extended - uncomment and set this to the display you
want to see
%the stim on - NOTE: PTB Will complain and interrupt unless you
include the
% line Screen('Preference', 'SkipSyncTests', 1)
% screenNumber = 1;

fixCrossDimPix = 10; %length in pixels of the arms of the
fixation cross - default 20

lineWidthPix = 2; % width in pixels of the arms of the fixation
cross - default 4

%% extract the trial order for the entered subject - determined
with run of designLumSets.m
load(lum.orderFile,'lumSet','patSet','nbackSet','audSet');

if lum.subjectNum ~=0
    lum.lumSet = lumSet(lum.subjectNum,:);

```

```

        lum.patSet = patSet(lum.subjectNum,:);
        lum.nbackSet = nbackSet(lum.subjectNum,:);
        lum.audSet = audSet(lum.subjectNum,:);
    else
        lum.lumSet = lumSet(lum.testSet,:);
        lum.patSet = patSet(lum.testSet,:);
        lum.nbackSet = nbackSet(lum.testSet,:);
        lum.audSet = audSet(lum.testSet,:);
    end

clear lumSet patSet nbackSet audSet;
%Initialize Psychtoolbox and Screens - not sure which, but
this combo
%removes the default bright white splash screen while loading
PTB
PsychDefaultSetup(2);
Screen('Preference', 'VisualDebugLevel', 1);
% Stop the program from quitting if the sync tests fail in
testing mode
% BE SURE TO REMOVE THIS AND TEST BEFORE PROPER EXPERIMENTS
% if lum.subjectNum == 0
%     Screen('Preference', 'SkipSyncTests', 1)
% end
Screen('Preference', 'DefaultFontSize', lum.textfont);
%redirect the typing location to the command window so that
calibration
%keypresses are not entered as changes in the code of this
script
commandwindow;

% Open a graphics window on the main screen
% using the PsychToolbox's PsychImaging function.
[window, windowRect]=PsychImaging('OpenWindow', screenNumber,
0,[],32,2);
Screen(window,'BlendFunction',GL_SRC_ALPHA,
GL_ONE_MINUS_SRC_ALPHA);
[width, height]=Screen('WindowSize', screenNumber);
[xCenter, yCenter] = RectCenter(windowRect);

ifi = Screen('GetFlipInterval', window);
numFrames = round(lum.numSecs / ifi);
numFramesBL = round(lum.baselinePupT / ifi);

% Coordinates for the fixation cross
xCoords = [-fixCrossDimPix fixCrossDimPix 0 0];

```

```

yCoords = [0 0 -fixCrossDimPix fixCrossDimPix];
allCoords = [xCoords; yCoords];

% Provide Eyelink with details about the graphics environment
% and perform some initializations.
el=EyelinkInitDefaults(window);

% We are changing calibration to a black background with white
targets,
% no sound and smaller targets
el.backgroundcolour = BlackIndex(el.window);
el.msgfontcolour = WhiteIndex(el.window);
el.imgtitlecolour = WhiteIndex(el.window);
el.targetbeep = 0;
el.calibrationtargetcolour = WhiteIndex(el.window);
el.calibrationtargetsize= .5;
el.calibrationtargetwidth=0.25;

% make changes to the calibration structure to take effect
EyelinkUpdateDefaults(el);

%% Initialization of the connection with the Eyelink
Gazetracker.

EyelinkInit(0);

% Navigate to the folder containing data files
cd(edfFilePath);

% Check to make sure that a file with this subjectNum has not
previously
% been collected (ensure previously collected data is not
overwritten)
if lum.subjectNum~=0 && isfile(edfFullName) == 0 %subjectnum not
set to test and an edffile for this subject does not exist
    i = Eyelink('Openfile', lum.edfFile);
    if i~=0 %0 means that the procedure worked - any other
number means a fail
        fprintf('Cannot create EDF file for ''%s'' ',
lum.edfFile);
        sca;
        Eyelink('Shutdown');
        return;
    end
elseif lum.subjectNum == 0 %test mode designated, will overwrite
last edf and *.mat files with SUB0 designation
    i = Eyelink('Openfile', lum.edfFile);

```

```

        if i~=0
            fprintf('Cannot create EDF file for ''%s'' ',
lum.edfFile);
            sca;
            Eyelink('Shutdown');
            return;
        end
    else
        fprintf('EDF file already exists for this subject!!\n Check
the SubjectNum variable for accuracy! ''%s'' ', edfFile);
        sca;
        Eyelink('Shutdown');
        return;
    end

%% SET UP TRACKER CONFIGURATION
% Setting the proper recording resolution, proper calibration
type,
% as well as the data file content;
Eyelink('command', 'add_file_preamble_text ''Recorded by
LumDynPupilFullV2 by CA''');
% This command is crucial to map the gaze positions from the
tracker to
% screen pixel positions to determine fixation
Eyelink('command','screen_pixel_coords = %ld %ld %ld %ld', 0, 0,
width-1, height-1);
WaitSecs(0.1); %need to give the host pc time to register
between successive commands
Eyelink('message', 'DISPLAY_COORDS %ld %ld %ld %ld', 0, 0,
width-1, height-1);
WaitSecs(0.1);
% set calibration type.
Eyelink('command', 'calibration_type = HV5');
WaitSecs(0.1);
% you must send this command with value NO for custom
calibration
% you must also reset it to YES for subsequent experiments
Eyelink('command', 'generate_default_targets = NO');
WaitSecs(0.1);
% STEP 5.1 modify calibration and validation target locations
Eyelink('command','calibration_samples = 6');
WaitSecs(0.1);
Eyelink('command','calibration_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command','calibration_targets = %d,%d %d,%d %d,%d %d,%d
%d,%d',...
        round(width/2),round(height/2),

```

```

round(width/2),round(height*0.25), round(width/2),round(height
- height*0.25), round(width*0.25),round(height/2), round(width
- width*0.25),round(height/2));
WaitSecs(0.1);
Eyelink('command','validation_samples = 6');
WaitSecs(0.1);
Eyelink('command','validation_sequence = 0,1,2,3,4,5');
WaitSecs(0.1);
Eyelink('command','validation_targets = %d,%d %d,%d %d,%d %d,%d
%d,%d',...
    round(width/2),round(height/2),
    round(width/2),round(height*0.25), round(width/2),round(height
- height*0.25), round(width*0.25),round(height/2), round(width
- width*0.25),round(height/2) );
WaitSecs(0.1);
% % set parser (conservative saccade thresholds)
% Eyelink('command', 'saccade_velocity_threshold = 35');
% WaitSecs(0.1);
% Eyelink('command', 'saccade_acceleration_threshold = 9500');
% WaitSecs(0.1);
% set EDF file contents - the data that will be recorded by the
host PC -
% DO NOT CHANGE!!
% remote mode possible add HTARGET ( head target)
Eyelink('command', 'file_event_filter =
LEFT,RIGHT, FIXATION, SACCADÉ, BLINK, MESSAGE, BUTTON, INPUT');
WaitSecs(0.1);
Eyelink('command', 'file_sample_data =
LEFT,RIGHT,GAZE,HREF,AREA,GAZERES,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% set link data - the values that we can query during the
recording
Eyelink('command', 'link_event_filter =
LEFT,RIGHT, FIXATION, SACCADÉ, BLINK, MESSAGE, BUTTON, FIXUPDATE, INPUT
');
WaitSecs(0.1);
Eyelink('command', 'link_sample_data =
LEFT,RIGHT,GAZE,GAZERES,AREA,STATUS,INPUT,HTARGET');
WaitSecs(0.1);
% calibration/drift correction target - controller not used but
removing
% this makes accepting fixation on the display pc KB no longer
work
Eyelink('command', 'button_function 5
"accept_target_fixation"');
WaitSecs(0.1);

```

```

% Hide the mouse cursor if testmode (sub0) not engaged - major
pain in the ass if
% the program fails - so only leave on when are certain code it
is working
if lum.subjectNum ~= 0
    Screen('HideCursorHelper', window);
end

% enter Eyetracker camera setup mode, calibration and validation
EyelinkDoTrackerSetup(el);
WaitSecs(1);
Eyelink('Command', 'set_idle_mode'); % seems to be a good
practice to ensure that the recording starts quickly
WaitSecs(0.5);
Eyelink('StartRecording');
WaitSecs(0.5); %there is a delay between this command is issues
and when eye data starts being collected - good to give it a few
hundred ms

%% Set up the stimuli and the luminance oscillations
%pull the relevent vals from the structure - too lazy to change
code from
%legacy usage
TLum = lum.lumRGBvalues(3);
MLum = lum.lumRGBvalues(2);
BLum = lum.lumRGBvalues(1);
%designated that the luminance vals were set manually -
distinguishes from
%earlier code (V1) that used pupil size to determin value - so
leave in for now
lum.lumDefaultUsed = [1 1 1];

%% Sets the variation amplitude for the sines
%amount to change luminance in each direction (in RGB)
ChngTLum = round(TLum*lum.LumChgPct);
ChngMLum = round(MLum*lum.LumChgPct);
ChngBLum = round(BLum*lum.LumChgPct);

%% make the sinusoidal luminance profiles
t = 0:ifi:lum.numSecs;
mdls1h = sin(2*pi*lum.lumChgFreqs(1)*t);
mdls02h = sin(2*pi*lum.lumChgFreqs(2)*t);
mdls01h = sin(2*pi*lum.lumChgFreqs(3)*t);

%% apply all sine profiles to 3 luminance levels
% 1 Hz - or whatever is set in lum.lumChgFreqs(1)

```

```

mdlsT1hRGB = mdls1h.*ChngTLum;
lumT1hRGB = mdlsT1hRGB+TLum;
lumT1h = lumT1hRGB./255; %put the RGB lum vals in PTB preferred
0-1 scale

mdlsM1hRGB = mdls1h.*ChngMLum;
lumM1hRGB = mdlsM1hRGB+MLum;
lumM1h = lumM1hRGB./255;

mdlsB1hRGB = mdls1h.*ChngBLum;
lumB1hRGB = mdlsB1hRGB+BLum;
lumB1h = lumB1hRGB./255;

% 0.2Hz - or whatever is set in lum.lumChgFreqs(2)
mdlsT02hRGB = mdls02h.*ChngTLum;
lumT02hRGB = mdlsT02hRGB+TLum;
lumT02h = lumT02hRGB./255;

mdlsM02hRGB = mdls02h.*ChngMLum;
lumM02hRGB = mdlsM02hRGB+MLum;
lumM02h = lumM02hRGB./255;

mdlsB02hRGB = mdls02h.*ChngBLum;
lumB02hRGB = mdlsB02hRGB+BLum;
lumB02h = lumB02hRGB./255;

%0.1Hz - or whatever is set in lum.lumChgFreqs(3)
mdlsT01hRGB = mdls01h.*ChngTLum;
lumT01hRGB = mdlsT01hRGB+TLum;
lumT01h = lumT01hRGB./255;

mdlsM01hRGB = mdls01h.*ChngMLum;
lumM01hRGB = mdlsM01hRGB+MLum;
lumM01h = lumM01hRGB./255;

mdlsB01hRGB = mdls01h.*ChngBLum;
lumB01hRGB = mdlsB01hRGB+BLum;
lumB01h = lumB01hRGB./255;

%% START THE DISPLAY PRESENTATION PHASE
%messages that appear on the screen before each trial - self
explanatory
adaptText = sprintf('%s','LIGHT ADJUSTMENT PERIOD');
adapttimeText = sprintf('%s%d%s', 'DURATION:
',lum.baselinePupMin,' minutes');
trialtimeText = sprintf('%s%d%s', 'DURATION: ',lum.numSecs,'

```

```

seconds');
keypressText = sprintf('%s','READ THE FIRST TWO LINES ABOVE, OUT
LOUD, WHEN YOU ARE READY TO START THIS TRIAL');

WaitSecs(0.1);
Eyelink('Message', 'LUM TRIALS START');
WaitSecs(0.1);

%% Loop for the running the sets - do not change

for set = 1:length(lum.lumSet)

    %message for instructing which n-back to do based on
predetermined
    %oder for that specific subject
    if lum.nbackSet(set) == 0
        nbackText = sprintf('%s','N-0 BACK');
    elseif lum.nbackSet(set) == 1
        nbackText = sprintf('%s','N-1 BACK');
    else
        nbackText = sprintf('%s','N-2 BACK');
    end
    %message for instructing which luminance profile it will do
based on predetermined
    %oder for that specific subject
    if lum.patSet(set) == 1
        patText = sprintf('%s','FIXED LUMINANCE');
    elseif lum.patSet(set) == 2
        patText = sprintf('%s','0.1 Hz LUMINANCE CHANGE');
    elseif lum.patSet(set) == 3
        patText = sprintf('%s','0.2 Hz LUMINANCE CHANGE');
    else
        patText = sprintf('%s','1 Hz LUMINANCE CHANGE');
    end
    %trial progress message
    trialnumText = sprintf('%s%d%s%d','TRIAL ',set,' out of
',length(lum.lumSet));
    %message inserted into the edffile before every trial to
indicate the condition set used
    Eyelink('Message', 'LUM %d PAT %d NBK
%d',lum.lumSet(set),lum.patSet(set),lum.nbackSet(set));

    %switch for determining luminance order
    switch lum.lumSet(set) %will do all trial conditions for
each luminance before switching to next

        case 1 % dimmest luminance set

```

```

        Screen('TextColor', window, [.7 .7 .7]);
        %sets adaptation to only run once at the start of
each
        %luminance phase
        if set == 1 || set == 13 || set == 25
            %makes the novel narration go in order
regardless of
            %luminance order
            if set == 1
                SoundSet =
sprintf('%s%s%d%s', soundpath, 'NovelReadingSoundPt1.mat');
            elseif set == 13
                SoundSet =
sprintf('%s%s%d%s', soundpath, 'NovelReadingSoundPt2.mat');
            else
                SoundSet =
sprintf('%s%s%d%s', soundpath, 'NovelReadingSoundPt3.mat');
            end
            %load the sound file (must be saved as *.mat)
            load(SoundSet, 'novelM', 'novelMfs');
            Screen('TextSize', window, lum.textfont);
            %sets corner square to black

            %sets visible window to frame 1 (or the mean
luminance for
            %the set - writes the appropriate screen
instructions as
            %determined earlier based on the trial set
conditions
            Screen('FillRect', window, [lumBlh(1) lumBlh(1)
lumBlh(1)]);
            Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
            round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%            Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('DrawText', window, adaptText,
lum.textleft, lum.texthigh(2));
            Screen('DrawText', window, adapttimeText,
lum.textleft, lum.texthigh(3));
            Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
            %show everything, tell the host PC that the

```

```

instructions
time before
    %are up, and then wait for minimum amount of
    %looking for a key press (any) to proceed
    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    KbStrokeWait;
    %play loaded sound - in this case the audiobook
file that
    %it loaded above
    sound(novelM,novelMfs);
    % this is the screen change loop - which is
important for
    % the change conditions - but not critical for
this - left
    % in for consistency
    for frame = 1:numFramesBL
        %make the trial square bright white

        %makes the visible area the appropriate
luminance for
        %each frame - in this case it is fixed, but
in later
        %loops this will change by frame number
        Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1) lumB1h(1)]);
        Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
        round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
        Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        %draws the fixation cross to the visible
area
        Screen('DrawLines', window, allCoords,...
            lineWidthPix, [1 0 0], [xCenter
yCenter], 2);
        %Shows the screen we just created, and if
this is the
        %first frame shown - places a stim on
message in the
        %edf file being recorded
        Screen('Flip', window);

```

```

        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
    % makes the corner square black again to
indicate trial is
    % done

    %sets the screen back to the mean luminance for
the set so
    %there is no black flickering that could cause a
plr
    Screen('FillRect', window, [lumBlh(1) lumBlh(1)
lumBlh(1)]);
    Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
    round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    %Shows the final screen and puts a message
marker in the
    %edf file indicating that this trial is over
    Screen('Flip', window);
    Eyelink('Message', 'STIM OFF');
    % clear sound stops playback (interrupts it if
the play
    % time is shorter than the set
    % the audio files -especially for the novel
narration are
    % really big, so it clears them once done to
free up
    % resources
    clear sound novelM novelMfs;
end
%This is the switch for the actual n-back trial
order
    switch lum.patSet(set)
        case 1 % N-Back Fixed Dim Luminance
            %loads the subject set determined random
number file
            %for the n-back task
            SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),

```

```

'.mat');

load(SoundSet,'randNum','randNfs');
Screen('TextSize', window, lum.textfont);

%visible square set to mean luminance for
this set - in
%this case, the dimmest luminance
Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1)]);
Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);

%corner square black - same as above
% Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
% make the instructions appropriate for the
task as is
% set by the subjectNum determined trial
order
Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
%shows what we just make and marks the edf
file
Screen('Flip', window);
Eyelink('Message', 'TEXT ON');
WaitSecs(lum.numPreSecs);
%wait on first frame for keypress from
experimenter
KbStrokeWait;
%plays the n-back random number sequence
previously
%loaded
sound(randNum,randNfs);
% loop for making the changing luminance
screens - this

```

```

                                % one is again fixed at a single luminance -
operates
                                % exacty as described above
                                for frame = 1:numFrames

                                    Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1)]);
                                    Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
                                round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
                                %
                                Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                                Screen('DrawLines', window,
allCoords,...
                                lineWidthPix, [1 0 0], [xCenter
yCenter], 2);
                                Screen('Flip', window);
                                if frame == 1
                                    Eyelink('Message', 'STIM ON');
                                end
                                end
                                % again - makes the last screen, black
corner square, and sends the stop
                                % message exactly as previously described

                                    Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1)]);
                                    Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
                                round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
                                Screen('Flip', window);
                                %
                                Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                                Eyelink('Message', 'STIM OFF');
                                % stops/interrupts playback and clears out
the audio file from memory
                                clear sound randNum randNfs;

                                case 2 % N-Back Slow Change Dim Luminance

```

```

        SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

        load(SoundSet,'randNum','randNfs');
        Screen('TextSize', window, lum.textfont);

        Screen('FillRect', window, [lumB01h(1)
lumB01h(1) lumB01h(1)]);
        Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
        round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
        Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
        Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
        Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
        Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
        Screen('Flip', window);
        Eyelink('Message', 'TEXT ON');
        WaitSecs(lum.numPreSecs);
        %wait on first frame for keypress from
experimenter

        KbStrokeWait;
        sound(randNum,randNfs);
        % this frame loop actually changes the
screen luminance

        % based on the frame order, otherwise
operates exactly

        % as the others perviously described
        for frame = 1:numFrames

                Screen('FillRect', window,
[lumB01h(frame) lumB01h(frame) lumB01h(frame)]);
                Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...

```

```

                                round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2))))];
%                                Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                                Screen('DrawLines', window,
allCoords,...
                                lineWidthPix, [1 0 0], [xCenter
yCenter], 2);
                                Screen('Flip', window);
                                if frame == 1
                                    Eyelink('Message', 'STIM ON');
                                end
                                end

                                Screen('FillRect', window, [lumB01h(1)
lumB01h(1) lumB01h(1)]);
                                Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
                                round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2))))];
%                                Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                                Screen('Flip', window);
                                Eyelink('Message', 'STIM OFF');
                                clear sound randNum randNfs;

                                case 3 % N-Back Medium Change Dim Luminance
                                    SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');
                                    load(SoundSet,'randNum','randNfs');
                                    Screen('TextSize', window, lum.textfont);

                                    Screen('FillRect', window, [lumB02h(1)
lumB02h(1) lumB02h(1)]);
                                    Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
                                    round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2))))];
%                                Screen('FillRect', window, [0 0 0],[0

```

```

round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4));
    Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
    Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
    Screen('DrawText', window, trialtimText,
lum.textleft, lum.texthigh(3));
    Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
    Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));

    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    %wait on first frame for keypress from
experimenter

    KbStrokeWait;
    sound(randNum,randNfs);

    for frame = 1:numFrames

        Screen('FillRect', window,
[lumB02h(frame) lumB02h(frame) lumB02h(frame)]);
        Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
        round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%        Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)];
        Screen('DrawLines', window,
allCoords,...
        lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end

    Screen('FillRect', window, [lumB02h(1)
lumB02h(1) lumB02h(1)]);
    Screen('FillRect', window, [0 0 0],...

```

```

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
    round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('Flip', window);
    Eyelink('Message', 'STIM OFF');

    clear sound randNum randNfs;

    case 4 % N-Back Fast Change Dim Luminance
        SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

        load(SoundSet,'randNum','randNfs');
        Screen('TextSize', window, lum.textfont);

        Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1)]);
        Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
    round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
    Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
    Screen('DrawText', window, trialtimText,
lum.textleft, lum.texthigh(3));
    Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
    Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));

    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    %wait on first frame for keypress from
experimenter

    KbStrokeWait;

```

```

        sound(randNum,randNfs);

        for frame = 1:numFrames

            Screen('FillRect', window,
[lumB1h(frame) lumB1h(frame) lumB1h(frame)]);
            Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
            round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
            Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('DrawLines', window,
allCoords,...
                lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

            Screen('Flip', window);
            if frame == 1
                Eyelink('Message', 'STIM ON');
            end
        end

        Screen('FillRect', window, [lumB1h(1)
lumB1h(1) lumB1h(1)]);
        Screen('FillRect', window, [0 0 0],...

[round(windowRect(3)*(lum.hedge(1)+lum.dimwid(1)))
round(windowRect(4)*(lum.vedge(1)+lum.dimwid(2)))...
            round(windowRect(3)*(lum.hedge(2)-
lum.dimwid(1))) round(windowRect(4)*(lum.vedge(2)-
lum.dimwid(2)))]);
%
            Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('Flip', window);
            Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;
    end

    case 2 %middle luminance set
        % Baseline Fixed Middle Luminance
        Screen('TextColor', window ,[1 1 1]);
        if set == 1 || set == 13 || set == 25

```

```

        if set == 1
            SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt1.mat');
        elseif set == 13
            SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt2.mat');
        else
            SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt3.mat');
        end
        load(SoundSet,'novelM','novelMfs');
        Screen('TextSize', window, lum.textfont);
        Screen('FillRect', window, [0 0 0]);
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumM1h(1) lumM1h(1)
lumM1h(1)],...
            [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
            round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawText', window, adaptText,
lum.textleft, lum.texthigh(2));
        Screen('DrawText', window, adapttimeText,
lum.textleft, lum.texthigh(3));
        Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
        Screen('Flip', window);
        Eyelink('Message', 'TEXT ON');
        WaitSecs(lum.numPreSecs);
        KbStrokeWait;

        sound(novelM,novelMfs);

        for frame = 1:numFramesBL
%            Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window, [lumM1h(1)
lumM1h(1) lumM1h(1)],...
                [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
                round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawLines', window, allCoords,...
                lineWidthPix, [1 0 0], [xCenter
yCenter], 2);
            Screen('Flip', window);

```

```

        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
    %
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumMlh(1) lumMlh(1)
lumMlh(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('Flip', window);
    Eyelink('Message', 'STIM OFF');

    clear sound novelM novelMfs;
end
%Middle luminance pattern switch
switch lum.patSet(set)
    case 1 % N-Back - Fixed Middle Luminance
        SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

        load(SoundSet,'randNum','randNfs');
        Screen('TextSize', window, lum.textfont);
        Screen('FillRect', window, [0 0 0]);
    %
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumMlh(1)
lumMlh(1) lumMlh(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
        Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
        Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
        Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
        Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));

        Screen('Flip', window);
        Eyelink('Message', 'TEXT ON');

```

```

WaitSecs(lum.numPreSecs);
%wait on first frame for keypress from
experimenter

KbStrokeWait;
sound(randNum,randNfs);

for frame = 1:numFrames
%
    Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumM1h(1)
lumM1h(1) lumM1h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('DrawLines', window,
allCoords,...
        lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

    Screen('Flip', window);
    if frame == 1
        Eyelink('Message', 'STIM ON');
    end
end
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumM1h(1)
lumM1h(1) lumM1h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('Flip', window);
    Eyelink('Message', 'STIM OFF');

clear sound randNum randNfs;

case 2 % N-back - Slow Change Middle Luminance
    SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

    load(SoundSet,'randNum','randNfs');
    Screen('TextSize', window, lum.textfont);
    Screen('FillRect', window, [0 0 0]);
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumM01h(1)

```

```

lumM01h(1) lumM01h(1)],...
    [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
    round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
    Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
    Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
    Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
    Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    %wait on first frame for keypress from
experimenter

    KbStrokeWait;
    sound(randNum,randNfs);

    for frame = 1:numFrames
%        Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window,
[lumM01h(frame) lumM01h(frame) lumM01h(frame)],...
            [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
            round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawLines', window,
allCoords,...
            lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
%    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumM01h(1)
lumM01h(1) lumM01h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...

```

```

        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('Flip', window);
    Eyelink('Message', 'STIM OFF');

    clear sound randNum randNfs;

    case 3 % N-Back - Medium Change Middle Luminance
        SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

        load(SoundSet,'randNum','randNfs');
        Screen('TextSize', window, lum.textfont);
        Screen('FillRect', window, [0 0 0]);
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumM02h(1)
lumM02h(1) lumM02h(1)],...
            [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
            round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
        Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
        Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
        Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
        Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));

        Screen('Flip', window);
        Eyelink('Message', 'TEXT ON');
        WaitSecs(lum.numPreSecs);
        %wait on first frame for keypress from
experimenter

        KbStrokeWait;
        sound(randNum,randNfs);

        for frame = 1:numFrames
%            Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window,
[lumM02h(frame) lumM02h(frame) lumM02h(frame)],...
                [round(windowRect(3)*lum.hedge(1))

```

```

round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawLines', window,
allCoords,...
        lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumM02h(1)
lumM02h(1) lumM02h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;

    case 4 % N-Back - Fast Change Middle Luminance

        SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

        load(SoundSet,'randNum','randNfs');
        Screen('TextSize', window, lum.textfont);
        Screen('FillRect', window, [0 0 0]);
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumM1h(1)
lumM1h(1) lumM1h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
        Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
        Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));

```

```

        Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
        Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
        Screen('Flip', window);
        Eyelink('Message', 'TEXT ON');
        WaitSecs(lum.numPreSecs);
        %wait on first frame for keypress from
experimenter

        KbStrokeWait;
        sound(randNum,randNfs);

        for frame = 1:numFrames
%
            Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window,
[lumM1h(frame) lumM1h(frame) lumM1h(frame)],...
[round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawLines', window,
allCoords,...
                lineWidthPix, [1 0 0], [xCenter
yCenter], 2);

            Screen('Flip', window);
            if frame == 1
                Eyelink('Message', 'STIM ON');
            end
        end
%
        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumM1h(1)
lumM1h(1) lumM1h(1)],...
[round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;
    end

    case 3 %bright lumin set
        Screen('TextColor', window ,[0 0 0]);
        % Baseline - Fixed Bright Luminance

```

```

        if set == 1 || set == 13 || set == 25
            if set == 1
                SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt1.mat');
            elseif set == 13
                SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt2.mat');
            else
                SoundSet =
sprintf('%s%s%d%s',soundpath,'NovelReadingSoundPt3.mat');
            end
            load(SoundSet,'novelM','novelMfs');
            Screen('TextSize', window, lum.textfont);
            Screen('FillRect', window, [0 0 0]);
%            Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window, [lumT1h(1) lumT1h(1)
lumT1h(1)],...
                [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
                round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawText', window, adaptText,
lum.textleft, lum.texthigh(2));
            Screen('DrawText', window, adapttimeText,
lum.textleft, lum.texthigh(3));
            Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
            Screen('Flip', window);
            Eyelink('Message', 'TEXT ON');
            WaitSecs(lum.numPreSecs);
            KbStrokeWait;

            sound(novelM,novelMfs);

            for frame = 1:numFramesBL
%                Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
                    [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
                    round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
                Screen('DrawLines', window, allCoords,...
                    lineWidthPix, [.5 0 0], [xCenter
yCenter], 2);

```

```

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
%       Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumT1h(1) lumT1h(1)
lumT1h(1)],...
            [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
            round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound novelM novelMfs;
    end

    switch lum.patSet(set)
        case 1 % N-Back - Fixed Bright Luminance
            SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

            load(SoundSet,'randNum','randNfs');
            Screen('TextSize', window, lum.textfont);
            Screen('FillRect', window, [0 0 0]);
%       Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
                [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
                round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
            Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
            Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
            Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
            Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
            Screen('Flip', window);
            Eyelink('Message', 'TEXT ON');

```

```

WaitSecs(lum.numPreSecs);
%wait on first frame for keypress from
experimenter

KbStrokeWait;
sound(randNum,randNfs);

for frame = 1:numFrames
%
    Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('DrawLines', window,
allCoords,...
        lineWidthPix, [.5 0 0], [xCenter
yCenter], 2);

    Screen('Flip', window);
    if frame == 1
        Eyalink('Message', 'STIM ON');
    end
end
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('Flip', window);
    Eyalink('Message', 'STIM OFF');

clear sound randNum randNfs;

case 2 % N-Back - Slow Change Bright Luminance
    SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

    load(SoundSet,'randNum','randNfs');
    Screen('TextSize', window, lum.textfont);
    Screen('FillRect', window, [0 0 0]);
%
    Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
    Screen('FillRect', window, [lumT01h(1)

```

```

lumT01h(1) lumT01h(1)],...
    [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
    round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
    Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
    Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
    Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
    Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
    Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    %wait on first frame for keypress from
experimenter

    KbStrokeWait;
    sound(randNum,randNfs);

    for frame = 1:numFrames

%           Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
           Screen('FillRect', window,
[lumT01h(frame) lumT01h(frame) lumT01h(frame)],...
           [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
           round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
           Screen('DrawLines', window,
allCoords,...
           lineWidthPix, [.5 0 0], [xCenter
yCenter], 2);

           Screen('Flip', window);
           if frame == 1
               Eyelink('Message', 'STIM ON');
           end
    end

%           Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
           Screen('FillRect', window, [lumT01h(1)
lumT01h(1) lumT01h(1)],...
           [round(windowRect(3)*lum.hedge(1))

```

```

round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;

        case 3 %N-Back - Medium Change Bright Luminance
            SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

            load(SoundSet,'randNum','randNfs');
            Screen('TextSize', window, lum.textfont);
            Screen('FillRect', window, [0 0 0]);
%            Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window, [lumT02h(1)
lumT02h(1) lumT02h(1)],...
                [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
                round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
            Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
            Screen('DrawText', window, trialtimText,
lum.textleft, lum.texthigh(3));
            Screen('DrawText', window, nbackText,
lum.textleft, lum.texthigh(2));
            Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));
            Screen('Flip', window);
            Eyelink('Message', 'TEXT ON');
            WaitSecs(lum.numPreSecs);
            %wait on first frame for keypress from
experimenter

            KbStrokeWait;
            sound(randNum,randNfs);

            for frame = 1:numFrames
%                Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
                Screen('FillRect', window,
[lumT02h(frame) lumT02h(frame) lumT02h(frame)],...
                    [round(windowRect(3)*lum.hedge(1))

```

```

round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawLines', window,
allCoords,...
        lineWidthPix, [.5 0 0], [xCenter
yCenter], 2);

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumT02h(1)
lumT02h(1) lumT02h(1)],...
        [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
        round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;

        case 4 % N-Back - Fast Change Bright Luminance
            SoundSet =
sprintf('%s%s%d%s',soundpath,'randNumsSoundSet',lum.audSet(set),
'.mat');

            load(SoundSet,'randNum','randNfs');
            Screen('TextSize', window, lum.textfont);
            Screen('FillRect', window, [0 0 0]);
%            Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
            Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
            [round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
            round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
            Screen('DrawText', window, trialnumText,
lum.textleft, lum.texthigh(1));
            Screen('DrawText', window, patText,
lum.textleft, lum.texthigh(4));
            Screen('DrawText', window, trialtimeText,
lum.textleft, lum.texthigh(3));
            Screen('DrawText', window, nbackText,

```

```

lum.textleft, lum.texthigh(2));
    Screen('DrawText', window, keypressText,
lum.textleft1, lum.texthigh(5));

    Screen('Flip', window);
    Eyelink('Message', 'TEXT ON');
    WaitSecs(lum.numPreSecs);
    %wait on first frame for keypress from
experimenter

    KbStrokeWait;
    sound(randNum,randNfs);

    for frame = 1:numFrames
%        Screen('FillRect', window, [1 1 1],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window,
[lumT1h(frame) lumT1h(frame) lumT1h(frame)],...
[round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('DrawLines', window,
allCoords,...
lineWidthPix, [.5 0 0], [xCenter
yCenter], 2);

        Screen('Flip', window);
        if frame == 1
            Eyelink('Message', 'STIM ON');
        end
    end
%        Screen('FillRect', window, [0 0 0],[0
round(windowRect(4)*.9) round(windowRect(3)*.1) windowRect(4)]);
        Screen('FillRect', window, [lumT1h(1)
lumT1h(1) lumT1h(1)],...
[round(windowRect(3)*lum.hedge(1))
round(windowRect(4)*lum.vedge(1))...
round(windowRect(3)*lum.hedge(2))
round(windowRect(4)*lum.vedge(2))]);
        Screen('Flip', window);
        Eyelink('Message', 'STIM OFF');

        clear sound randNum randNfs;

    end
end
end

%% WRAP UP AND FINAL PROCEDURES

```

```

%Make the end screen text with the thank you note
endText1 = sprintf('%s','YOU HAVE COMPLETED THE EXPERIMENT!');
endText2 = sprintf('%s','THANK YOU FOR PARTICIPATING!');

%black out screen to indicate session end
Screen('TextColor', window ,[1 1 1]);
Screen('FillRect', window, [0 0 0],...
    [round(windowRect(3)*.2) round(windowRect(4)*.2)...
    round(windowRect(3)*.8) round(windowRect(4)*.8)]);
Screen('DrawText', window, endText1, lum.textleft,
lum.texthigh(2));
Screen('DrawText', window, endText2, lum.textleft,
lum.texthigh(3));
Screen('Flip', window);
%plant a final message in the edf file to indicate that the
session
%concluded completely - useful for detecting if the set was
interrupted
%after the fact
Eyelink('Message', 'SESSION END');
%Buffer time to ensure the recording stops well after the
collection period
%concludes and the completion message is written
WaitSecs(0.5);
%Stop the eye tracker recording data
Eyelink('StopRecording');
%more buffer the let that operation complete
WaitSecs(0.5);
%again - seems to be preffered practice - not entirely certain
why
Eyelink('Command', 'set_idle_mode');
WaitSecs(0.5);
%closes the file and seals the deal
Eyelink('CloseFile');
WaitSecs(0.5);

%% Data Transfer to host PC and S#Data.mat file saving
% go to the location for the file to be saved
cd(edfFilePath);
%make the mat file save name and save the structure 'lum' which
contains all of the
%session details needed to reconstruct the exact procedure
lum.edfFilename = sprintf('%sData.mat',lum.edfFile);
save(lum.edfFilename,'lum');

% This transfers the data file from the host PC into the

```

```

directory we
% specified in the beginning.  They reccomend leaving this in a
try loop
% so that the tracker still gets the link shutdown command in
this transfer
% somehow fails
try
    fprintf('Receiving data file ''%s''\n', lum.edfFile );
    status=Eyelink('ReceiveFile');
    if status > 0
        fprintf('ReceiveFile status %d\n', status);
    end
    if 2==exist(lum.edfFile, 'file')
        fprintf('Data file ''%s'' can be found in ''%s''\n',
lum.edfFile, pwd );
    end
catch
    fprintf('Problem receiving data file ''%s''\n', lum.edfFile
);
end
%if the transfer fails, the edf file is still saved, it will
just have to be manually
%collected from the /etc/data/ folder on the host pc using the
file manager
WaitSecs(0.1);
%Closes the link between the display and host PC so that it is
ready for
%the next experiment
Eyelink('Shutdown');
%closes all of the PTB created screens that may still be open
Screen('CloseAll');

end

```

U.S. Army Aeromedical Research Laboratory Fort Rucker, Alabama

All of USAARL's science and technical
information documents are available for
download from the
Defense Technical Information Center.

<https://discover.dtic.mil/results/?q=USAARL>



**Army Futures Command
U.S. Army Medical Research and Development Command**