Utilization of Automated Eye-Tracking as an Ancillary Diagnostic Test in Neurological Disorders

By

Hadi Karimi

Submitted in partial fulfillment of the academic requirements for the degree of Master of Medicine in the Department of Neurology

School of Clinical Medicine, College of Health Sciences

University of KwaZulu-Natal

Durban, South Africa, 2024

As the candidate's supervisor I have approved this thesis for submission.

Signed: November 25, 2024

[redacted]

Prof Anandan A. Moodley

Declaration

I “Hadi Karimi” declare that;

1) The research reported in this dissertation, except where otherwise indicated, is my original work.

2) This dissertation has not been submitted for any degree or examination at any other university.

3) This dissertation does not contain other persons’ data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

4) This dissertation does not contain other persons’ writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

    a) their words have been re-written but the general information attributed to them has been referenced;

    b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

5) Where I have reproduced a publication of which I am an author, co-author or editor, I have indicated in detail which part of the publication was actually written by myself alone and have fully referenced such publications.

6) This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References sections.

Signed: November 25, 2024

Authorship Statement and Acknowledgements

Authors

This study was conducted under the collaborative efforts of Dr. Karimi (Author and Investigator) and Prof Anand Moodley (Supervisor), with occasional input from Prof. Vinod Patel (Head of department of Neurology). Authorship contributions are detailed below in accordance with the CRediT (Contributor Roles Taxonomy) framework:

Dr. Karimi (Author and Investigator):

- Data Curation: [Lead] literature review and background.
- Conceptualization: [Equal] developing Hypothesis, Aims, Objectives, and Outcome.
- Methodology: [Lead] planning, crafting the methodology, and developing testing protocols.
- Software: [Lead] developing an application with analytic software and supporting algorithms.
- Investigation: [Lead] conducting and executing experiments, and data collection.
- Formal Analysis: [Lead] post hoc analysis, statistical analysis, result interpretation, clinical impact, case presentation and reporting.
- Writing – Original Draft: [Lead] drafting the manuscripts, designing tables, graphs, and figures, abstracts, poster and presentation.
- Visualization: [Lead] producing poster and presentation.

Prof Anand Moodley (Supervisor):

- Conceptualization: [Lead] providing the original intellectual concept for optimizing eye-tracking testing in neurological disorders. [Equal] refining developing Hypothesis, Aims, Objectives, and Outcome.
- Methodology: [Support] guidance in periodic consultations to review methodology and monitor project progress.
- Validation: [Equal] clinical insights into the clinical interpretation of qualitative results particularly for patient group.
- Writing – Review & Editing: [Support] reviewing and recommending improvements to manuscripts, posters, and presentations.

Prof Vinod Patel (Head of department of Neurology):

- Project Administration: [Support] providing insights to optimize the project's foundation for future, more focused research initiatives.

Use of Editing Tools and Artificial Intelligence (AI)

The manuscript was refined using professional, non-generative editing tools to enhance linguistic clarity, professionalism, and style, while preserving the original user input and without altering the originality of the intellectual insights. The following tools were utilized:

- **Microsoft Editor Professional (AI- powered):** to enhance grammar, punctuation, style, clarity, readability.
- **Grammarly Academic Edition (AI-powered):** to enhance sentence structure, phrasing, academic tone, clarity, and coherence.
- **Scribbr's Online (AI-powered):** to improve phrasing, fluency and clarity.

No generative AI tools, including Long Language Model (LLM)-powered systems, were used to generate content or computer programming code. All content and analyses, except where explicitly stated, reflect the original work of the authors.

Resources and Financial Assistance

The Department of Neurology provided the eye-tracker machine, workstation computer, accessories, and facilities necessary for this research. No additional financial assistance was received for this study.

Acknowledgements

The author extends heartfelt gratitude to Professor Anandan Moodley for his invaluable mentorship, marked by guidance and camaraderie that enriched every step of this journey. Special thanks are also owed to Professor Vinod Patel and the entire Department of Neurology at UKZN, whose support and resources were instrumental in bringing this work to fruition.

Dedication

To the decent, compassionate, and caring Sara.

Abstract

Modern eye-tracking technology, a non-invasive modality that records eye movements in response to visual stimuli, has shown significant promise as a diagnostic tool for neurological disorders. Its capacity to detect subtle abnormalities in ocular motor function, often linked to underlying neural circuitry dysfunction, provides a distinct advantage over traditional bedside examinations, which may overlook these nuances and typically require more advanced or invasive diagnostic modalities. Moreover, these advanced methods are not always readily available in routine clinical practice and may not capture the valuable insights that eye-tracking can provide.

Despite its potential, the application of eye-tracking has largely remained confined to research settings, often restricted by a narrow focus on specific neurological conditions and a reliance on individualized data analysis. This has impeded its broader adoption in clinical neurology, where standardized, scalable, and automated diagnostic tools are essential.

In this project, we aimed to bridge this gap by developing a comprehensive and automated Diagnostic Eye-Tracking Study (DETS) tailored for routine clinical practice. We established standardized protocols incorporating relevant visual stimuli and tasks, and defined biometrics that are both informative and practical for clinical neurology. Additionally, we developed an advanced analytical software platform capable of generating both quantitative and qualitative results, ensuring robust and clinically relevant interpretations.

To validate our approach, we conducted a pilot study that demonstrated the efficacy and feasibility of the protocols, as well as the validity and reliability of our automated analytical software. The study confirmed that our approach offers a valuable ancillary tool to enhance diagnostic accuracy across a range of neurological disorders. By integrating this tool into standard neurological assessments, we anticipate a significant improvement in patient outcomes.

Contents

Figures and Tables

Abbreviations

AI	Artificial Intelligence
AOS	Accessory Optic System
BREC	Biomedical Research Ethics Committee
CRediT	Contributor Roles Taxonomy
DETS	Diagnostic Eye-Tracking Study
DLPN	Dorsolateral Pontine Nucleus
EBN	Excitatory Burst Neuron
EEG	Electroencephalography
EOG	Electrooculography
FEF	Frontal Eye Field
fMRI	Functional Magnetic Resonance Imaging
HIV	Human Immunodeficiency Virus
IBN	Inhibitory Burst Neuron
IEEE	Institute of Electrical and Electronics Engineers
L	Left
LLM	Long Language Model
LogMAR	Logarithm of the Minimum Angle of Resolution
MLF	Medial Longitudinal Fasciculus
MMSE	Mini–Mental State Examination
MRI	Magnetic Resonance Imaging
MST	Medial Superior Temporal
OKN	Optokinetic Nystagmus
PPRF	Paramedian Pontine Reticular Formation
PSP	Progressive Supranuclear Palsy
R	Right
RGB	Red Green Blue
riMLF	Rostral interstitial nucleus of the Medial Longitudinal Fasciculus
RMS	Root Mean Square
RMSE	Root Mean Square Error
SCA	Spinocerebellar Ataxia
SNR	Signal-to-Noise Ratio
TEF	Temporal Eye Field
VOR	Vestibulo-Ocular Reflex

Introduction

The extensive anatomical and functional distribution of eye movement circuitry throughout the nervous system results in eye movement aberrations being prevalent in a wide range of neurological disorders.[1] This omnipresent involvement of ocular motor control underscores the important role of eye movement aberrations as indicators of underlying neurological conditions (*Table 1*). Detecting and evaluating specific patterns of these abnormalities, in conjunction with clinical features, can significantly enhance our understanding of various neurological disorders.[2,3]

However, the gross bedside oculomotor examination, a routine component of neurological assessment, relies heavily on subjective observation, which is contingent upon the examiner's expertise and experience.[4] Consequently, many neurological disorders that do not directly and substantially affect the oculomotor system may not present with overt eye movement abnormalities but rather with subtle aberrations that may not be promptly captured through observation.[5] Furthermore, bedside examination fails to objectively quantify and qualify the physical characteristics of these aberrations, emphasizing the potential of high-resolution ocular motility assessments as a supplemental diagnostic tool to reveal disorder-specific characteristics of eye movement abnormalities in neurological disorders.[6]

Recent advancements in eye-tracking technology have facilitated the precise, objective, and reproducible measurement of eye movement aberrations.[7] Compared to other advanced diagnostic modalities, modern eye-tracking systems are non-invasive, less resource-intensive, and do not require specialized facilities, making them suitable for use in various settings, including neurology clinics.[8] Recent studies have highlighted the diagnostic potential of eye-tracking technology across a range of neurological disorders. For instance, eye-tracking has shown promise in the early detection of movement disorders, identifying subtle eye movement changes that precede motor symptoms in Parkinson's disease, thus allowing for earlier intervention and monitoring before significant neuronal loss occurs.[9–15]

Similarly, specific patterns of saccadic and pursuit impairments have been shown to correlate with disease severity and cognitive decline in neurodegenerative conditions such as Alzheimer's disease.[16–18] Eye-tracking has also been utilized to enhance stroke diagnosis by quantifying eye movements to detect subtle abnormalities that may be missed in early imaging, highlighting its potential utility in acute neurological assessments.[19,20] Additionally, eye-tracking has been employed to distinguish between different types of ataxias, revealing distinct eye movement abnormalities among spinocerebellar ataxia subtypes, further supporting its role as a non-invasive biomarker for differential diagnosis.[21,22]

The advancements and advantages in eye-tracking, combined with growing evidence of its ability to enhance diagnostic accuracy, prognostication, disease progression monitoring, and treatment efficacy across various neurological disorders, reinforce its potential as a valuable addition to

existing diagnostic modalities in clinical practice. Hence, expanding its application beyond current application, limited to research settings, could provide clinicians with objective data to improve decision-making processes.

However, despite its promising potential, several limitations currently hinder the widespread adoption of eye-tracking in routine clinical practice. These limitations include the lack of standardized protocols with clinically relevant biometrics and corresponding normative data, as well as the recognition of characteristic-specific patterns for various neurological conditions.[8,23] Addressing these challenges will require the development of standardized protocols and automated analytical software tailored to clinical neurology.

Current eye-tracking software solutions provided by system vendors are primarily designed for marketing and behavioral studies.[24] For example, platforms like Tobii® Pro Lab offer eye-tracking solutions that mainly focus on user interactions with presented content, generating heatmaps that highlight areas of high visual attention.[25] While such software are useful for general behavioral studies, they do not meet the nuanced requirements needed for analyzing neurological disorders, which often involve subtle and complex eye movement abnormalities. Despite the high precision and sampling rates offered by eye-tracking hardware, these software solutions lack the necessary granularity and precision for processing and analyzing data in clinical settings.[26,27]

Studies investigating eye-tracking in neurological disorders often employ custom-made analytic software developed by research teams for specific study purposes. These systems are tailored to the unique hypotheses and methodologies of each study, leading to considerable variability in design, limited functionality, and a lack of comprehensive standardized testing protocols.[28,29] Additionally, these solutions often require manual intervention for case-based analysis, lacking the automation and scalability necessary for widespread clinical application. As a result, they are not suitable for automated mass testing, limiting their use to controlled research settings rather than routine clinical practice.[30,31]

Aims and Objectives

The aim of this project was to adopt and utilize eye-tracking as a routine diagnostic modality in clinical neurology akin to nerve conduction study.

The primary objectives of this project were to:

1. Provide an overview of eye-tracking to evaluate eye movements.
2. Develop a comprehensive battery of eye-tracking protocols with appropriate visual stimuli for fixation, saccades, pursuit, optokinetic nystagmus, prosaccades, and antisaccades.
3. Define protocol specific informative biomarkers pertinent to various neurological conditions.
4. Develop analytical software capable of implementing advanced signal processing algorithms.
5. Validate the software and obtain normative data through a pilot study.
6. Present qualitative and quantitative data of selected cases that illustrate the benefit of eye tracking to clinical neurology

Table 1: Eye movement abnormalities in neurological disorders.

CATEGORY	DISORDER	PRESENTATION
Degenerative [10,16,17,32 40]	Progressive Supranuclear Palsy	Supranuclear vertical gaze palsy, reduced vertical saccadic velocity, impaired vertical smooth pursuit
	Parkinson's Disease	Reduced saccadic gain (hypometric) and increased latency, impaired smooth pursuit
	Multiple System Atrophy	Reduced saccadic velocity, gaze-evoked nystagmus
	Corticobasal Degeneration	Oculomotor apraxia, asymmetric gaze palsy
	Alzheimer's Disease	Fixation instability, increased saccadic latency, increased antisaccades error without correction
	Dementia with Lewy bodies	Increased saccadic latency, increased prosaccades and antisaccades error
	Frontotemporal Dementia Behavioural	Increased antisaccades error with correction
	Frontotemporal Dementia Semantic	Normal antisaccades
Brainstem [5,41 44]	Internuclear Ophthalmoplegia	Gaze palsy with nystagmus
	Brainstem Stroke	Gaze-evoked direction changing nystagmus
	Parinaud's Syndrome	Upward gaze palsy, convergence-retraction nystagmus
Cerebellar [19,21,22,43 46]	Friedreich's Ataxia	Dysmetric saccades, gaze-evoked nystagmus
	Spinocerebellar Ataxias	Saccadic intrusions, ocular dysmetria
	Cerebellar Stroke	Gaze-evoked nystagmus, dysmetric saccades, ocular dysmetria
	Ataxia-Telangiectasia	Oculomotor apraxia, impaired smooth pursuit, dysmetric saccades
Basal Ganglia [47 51]	Huntington's Disease	Increased saccadic latency, decreased saccadic gain, reduced saccadic velocity, impaired fixation
	Wilson's Disease	Impaired smooth pursuit, dysmetric saccades
Cortical [52 56]	Parietal Lobe Lesions	Impaired reflexive saccades
	Frontal Lobe Lesions	Impaired task specific saccades, increased prosaccades and/or antisaccade errors
	Occipital Lobe Lesions	Impaired smooth pursuit
Autoimmune [57 61]	Myasthenia Gravis	Saccadic velocity decrements (fatigable), increased smooth pursuit phase shift
	Multiple Sclerosis	Internuclear ophthalmoplegia, nystagmus
	Neuromyelitis Optica	Gaze-evoked nystagmus
Metabolic [62,63]	Wernicke's Encephalopathy	Horizontal gaze palsy, nystagmus
Paraneoplastic [64,65]	Opsoclonus-Myoclonus Syndrome	Opsoclonus, nystagmus, saccadic dysmetria, impaired smooth pursuit
Infectious [66 68]	HIV Encephalopathy	Saccadic dysmetria, impaired smooth pursuit
Traumatic [69,70]	Traumatic Brain Injury	Impaired smooth pursuit
Genetic [14,47]	Niemann-Pick Disease Type C	Vertical supranuclear gaze palsy
	Tay-Sachs Disease	Reduce saccadic velocity

Background

Neuroscience

Understanding the neuroscience of eye movement and their characteristics is fundamental for assessing and diagnosing abnormalities (*Table 2*). Functionally, conjugate eye movements can be grouped into two broad systems: gaze stabilization and gaze shifting.[1]

Table 2: Summary of Eye Movement Characteristics.

CLASS	FUNCTION	VELOCITY	LATENCY	TRAJECTORY
Foveation [71]	Maintain focus on a stationary target	N/A	N/A	Small oscillations
Saccade [1,72,73]	Rapidly shift gaze to a new target	100-600 degrees/s (amplitude dependent)	150-250 milliseconds	Straight
Pursuit [74 77]	Track a moving target smoothly	30-100 degrees/s (target speed dependent)	N/A	Smooth
Vergence [78,79]	Adjust focus for targets at varying distances	15 degrees/s	150-250 milliseconds	Disconjugate
Vestibulo–ocular reflex (VOR) [1,80,81]	Stabilize gaze during head movement	< 500 degrees/s (head acceleration dependent)	< 15 milliseconds (subject to adaptation and decay)	Compensatory
Optokinetic Nystagmus (OKN) [82 84]	Stabilize gaze during visual felid movement	< 100 degrees/s (retinal slip dependent)	50-100 milliseconds	Compensatory
Nystagmic fast phase [84]	Reposition eyes during continuous VOR or OKN	500 degrees/s	N/A	Quick reset
Microsaccades [85 87]	Correct minor fixation errors	1-20 degrees/s	20-100 milliseconds	Small and rapid
Fixational Tremor [88,89]	Fine-tune precise fixation	30-100 Hz, very small amplitude	Immediate	High frequency

Gaze Stabilization

Fixation (Foveation)

When both the object of interest and the head are static, minuscule eye movements such as tremor, drift, and microsaccades enhance visual acuity by optimizing the quality of light received by foveal photoreceptors, thereby ensuring better spatial resolution for cortical image processing.[71,90] Color-specific retinal photoreceptors (L-cone, M-cone, S-cone) respond most effectively when exposed to a coherent and well-defined spectrum of frequencies within their activation range.[91] However, these photoreceptors quickly undergo desensitization (fatigue), making the small eye movements essential to prevent continuous activation by the same light ray.[85,92] Additionally, microsaccades enhance contrast sensitivity by engaging more rod receptors as well as correcting physiological drifts that shift the image off the fovea, a consequence of the tonic activity of extraocular muscles.[93]

Oculomotor neurons exhibit characteristic burst, tonic, and inhibitory activities, leading to pulse and step actions. The pulse initiates eye movement by driving muscle contraction, while the step maintains the eye's position through sustained muscle tonicity. Simultaneously, inhibitory neurons suppress activity in the antagonist muscles. These actions are coordinated by brainstem structures, including the nucleus prepositus hypoglossi for horizontal eye movements and the interstitial nucleus of Cajal for vertical movements (*Figure 1*). [1,94]

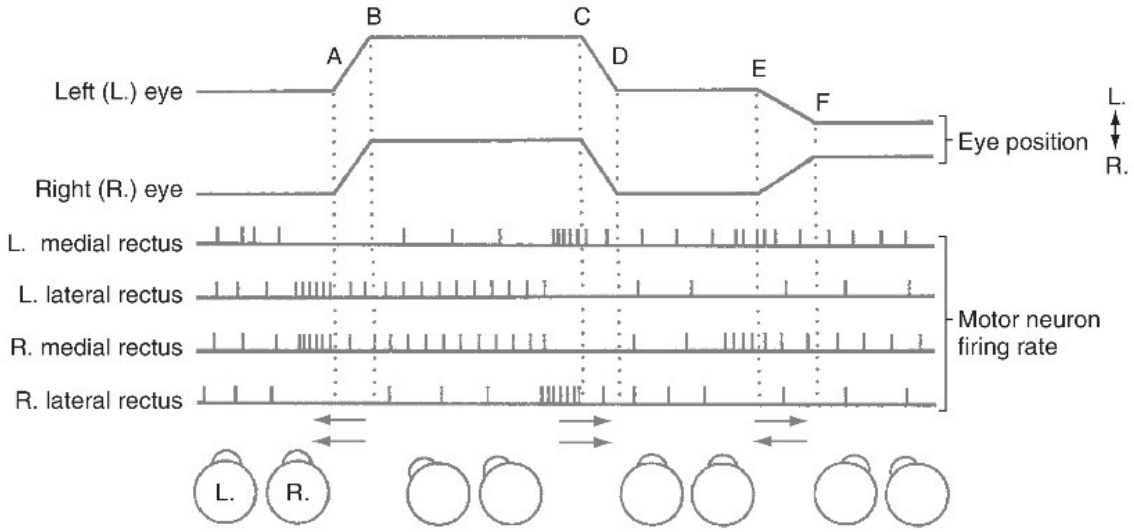


Figure 1: Motor neuron firing patterns for horizontal saccades and vergence movements. [95]

Dysfunctions in gaze stabilization typically manifest as intrusions during fixation. For example, jerk nystagmus is commonly associated with peripheral vestibular dysfunction, while central vestibular dysfunction, such as that resulting from brainstem or cerebellar lesions (e.g., pontine strokes), can present with pure vertical or torsional jerk nystagmus.[96] Pendular nystagmus is

frequently observed in congenital disorders like albinism, and in acquired conditions involving demyelination in the brainstem or cerebellum, such as multiple sclerosis.[97] Gaze-evoked nystagmus can be due to lesions in the cerebellum, particularly in the flocculus and paraflocculus, which disrupt the neural integration necessary to maintain eccentric gaze positions. This form of nystagmus can also indicate cerebellar degenerative diseases such as spinocerebellar ataxia, acute cerebellar strokes, or drug toxicity (e.g., phenytoin, carbamazepine).[98]

Downbeat nystagmus is often associated with disorders affecting the craniocervical junction, such as Chiari malformation, or with lesions in the cerebellar flocculus.[99] Upbeat nystagmus can be linked to lesions in the medulla or midbrain, as seen in conditions like Wernicke's encephalopathy.[100] Additionally, see-saw nystagmus may be observed in parasellar lesions affecting the optic chiasm, such as pituitary tumors.[101] Furthermore, saccadic intrusions and oscillations during fixation, including square-wave jerks, ocular flutter, and opsoclonus, are commonly seen in cerebellar disease, demyelinating conditions, or paraneoplastic syndromes.[102]

Vestibulo-Ocular Reflex

These corrective eye movements hold the image steady during brief or rapid head movement while the field of view remains static. The Vestibulo-ocular reflex is mediated by the semicircular canals and otolith organs of the inner ear, which detect head motion and send signals via the vestibular nuclei to the oculomotor nuclei to stabilize gaze.[1]

Optokinetic Eye Movement

When the head is static but the entire visual field moves across the retina (retinal slip), compensatory eye movements are triggered to stabilize the image on the retina. These movements, known as optokinetic nystagmus (OKN), are driven by changes in the tonic firing rate of oculomotor neurons that move the eyes in the direction of the visual field movement, followed by burst firing in neurons that move the eyes in the opposite direction, causing a rapid return to the primary position.[1] The accessory optic system (AOS) initiates these movements when retinal slip is detected; otherwise, the foveation system overrides the optokinetic response (*Figure 2*).[103,104] The gain of OKN is controlled by the cerebellum, similar to vestibulo-ocular reflexes, so lesions in this pathway can lead to significant deficits in both reflexes.[105]

Clinically, the OKN response can be elicited by presenting a succession of moving visual stimuli at angles of 5 to 10 degrees. Responses in one direction are compared with those in the opposite direction. Asymmetric OKNs may indicate parietal lobe pathology, such as a mass-occupying lesion, whereas partial field deficits from occipital lobe lesions typically do not affect OKN. OKN assessment can also reveal slowed adducting saccades in subtle cases of internuclear ophthalmoplegia and accentuate nystagmus in the abducting eye. Additionally, OKN abnormalities may appear early in the course of progressive supranuclear palsy.[105 107]

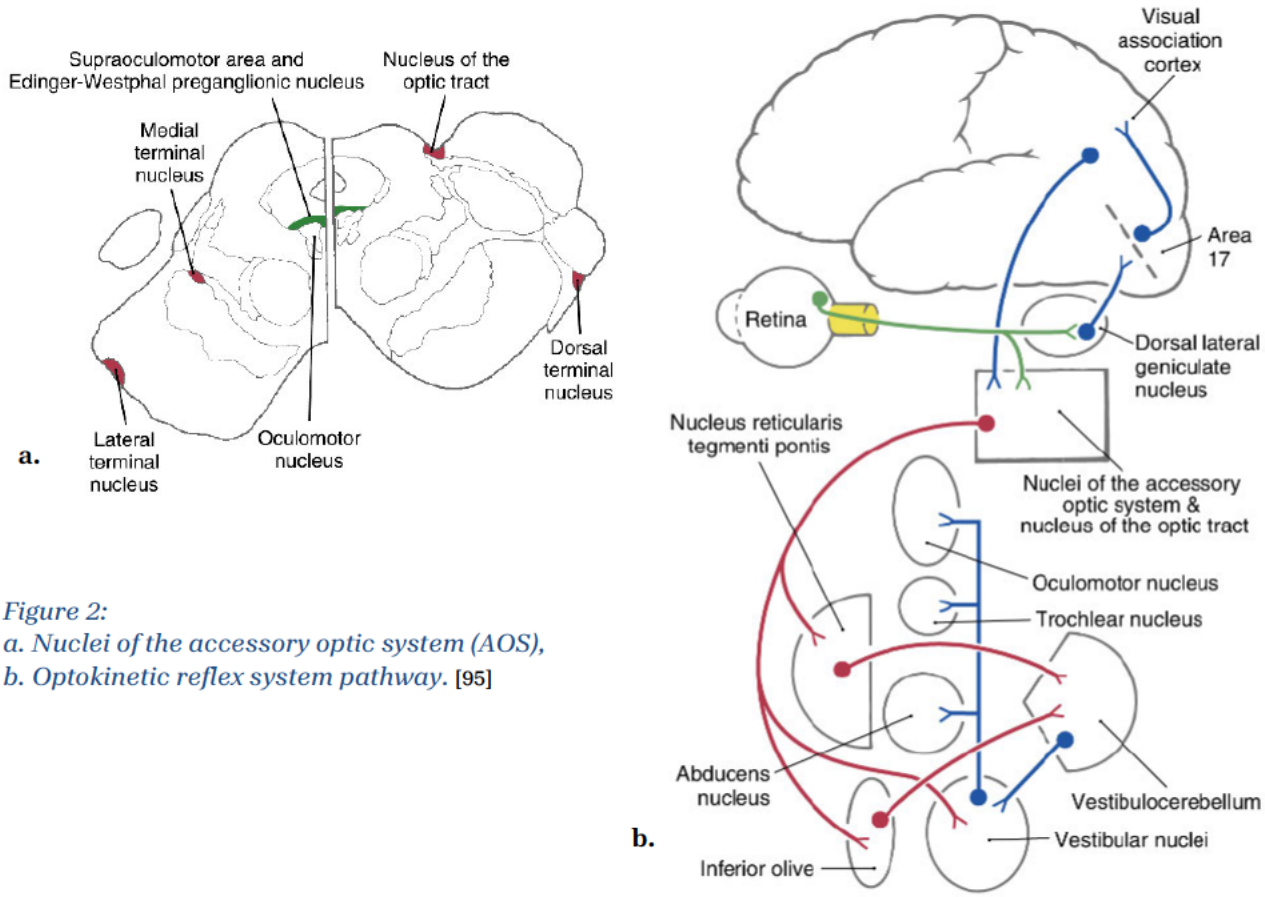


Figure 2:
a. Nuclei of the accessory optic system (AOS),
b. Optokinetic reflex system pathway. [95]

Vergence

When the visual cortex receives binocular images from the retina, a high degree of retinal disparity (the difference in the points on each retina where an image falls) increases the firing rate of motor neurons via the supraoculomotor area in the midbrain. This response adjusts the angle between the eyes to fuse the images, accommodating for target distance. These disconjugate eye movements, which include convergence or divergence, are typically slow reflexive saccades and are coupled with changes in lens curvature and pupil size to ensure proper foveation.[108]

Gaze Shifting

Saccadic Movements

Saccades are brief, rapid eye movements that redirect the eyes toward visual targets. During a saccade, the visual system suppresses incoming visual input, making these movements generally unnoticeable.[1,109] The visual association cortex ensures visual constancy by integrating information from each fixation into a seamless representation of the visual world. The superior colliculus plays a key role in guiding saccades by integrating visual, auditory, and somatosensory information, while the cerebellum fine-tunes saccadic accuracy and provides adaptive control.[110,111]

Saccades Circuitry

The frontal eye field (FEF) generates command signals to initiate eye movements in the contralateral direction. Neurons in the FEF calculate the direction and amplitude of these movements, sending motor signals to the caudate nucleus and the superior colliculus via the internal capsule, crus cerebri, and corticotectal tract. The superior colliculus then relays these signals to the vertical and horizontal gaze centers located in the midbrain and paramedian pontine reticular formation (PPRF), respectively (Figure 3).[112]

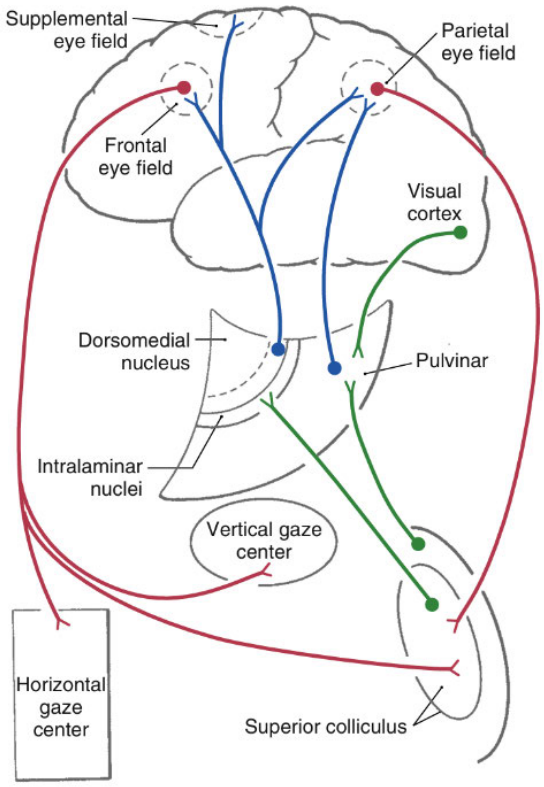


Figure 3: Supranuclear Control.[95]

The vertical gaze center, located in the midbrain reticular formation, controls the lower motor neurons in the oculomotor and trochlear nuclei. The horizontal gaze center, located in the PPRF, controls the lower motor neurons in the ipsilateral abducens nucleus and, via the contralateral medial longitudinal fasciculus, controls the contralateral oculomotor nucleus. The superior colliculus receives afferent inputs from the retina, inferior colliculus (which processes auditory information), visual association areas of the parietal lobe, and basal ganglia. These inputs allow the superior colliculus to modulate the signals that determine the amplitude and direction of saccades (*Figure 4*).[113,114]

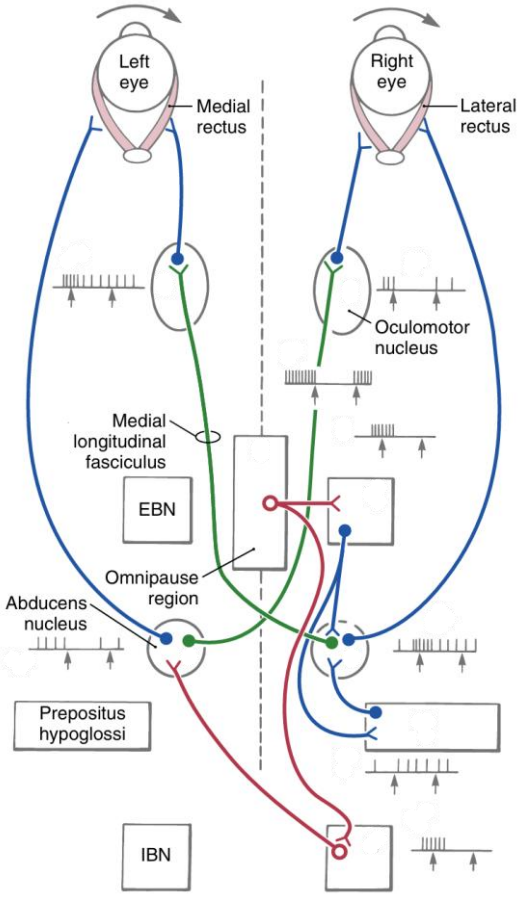


Figure 4: Horizontal saccades pathway. EBN, Excitatory Burst Neuron Area; IBN, Inhibitory Burst Neuron Area.[95]

Basal ganglia structures, including the caudate nucleus and substantia nigra, modulate the activity of the superior colliculus through direct and indirect pathways. The caudate nucleus receives excitatory input from the FEF and sends inhibitory signals to the substantia nigra. The substantia nigra, in turn, sends inhibitory signals to the superior colliculus. However, this inhibition is controlled by the direct input from the caudate nucleus. The role of the substantia nigra pars reticulata in inhibiting the superior colliculus is essential for the suppression and voluntary control of saccades (*Figure 5*).[115,116]

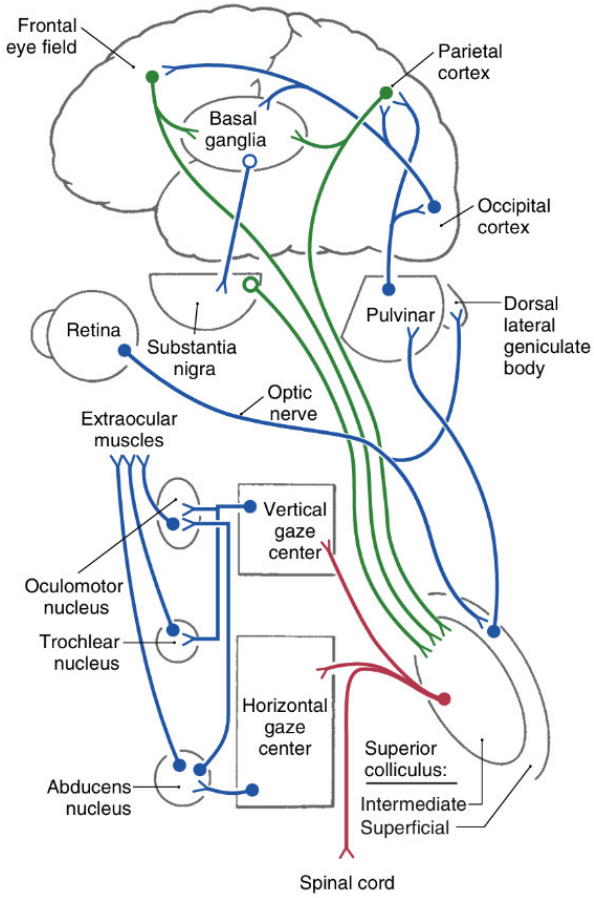


Figure 5: Superior colliculus pathways. [95]

The superior colliculus sends axons to the gaze centers via the tectospinal tract and can initiate and control saccades independently of the FEF.[117] While the FEF primarily initiates voluntary and memory-guided (task-specific) saccades, the superior colliculus is responsible for initiating reflexive saccades. Voluntary saccades are typically not triggered by visual stimuli; instead, visual control occurs after the eye movement.[115] The posterior parietal visual association cortex, part of the dorsal visual stream, assesses whether the visual target has been successfully acquired and sends corrective signals to the FEF and superior colliculus if the target is not in view. Additionally, the FEF plays a role in predicting and anticipating target motion, which is essential for initiating predictive saccades.[114]

Saccadic dysfunction can result from lesions in the FEF, parietal cortex, superior colliculus, cerebellum, brainstem, and basal ganglia. Lesions in the FEF typically cause a transient inability to generate contralateral saccades, as observed in acute strokes affecting the FEF. In contrast, chronic FEF lesions may lead to increased latency and decreased accuracy of voluntary saccades. Damage to the parietal cortex, particularly the intraparietal sulcus, can impair the initiation and accuracy of visually guided saccades. This is exemplified in Balint's syndrome, where patients have difficulty shifting gaze to visual stimuli, a condition known as oculomotor apraxia.[34,47,113,118]

Lesions in the superior colliculus can lead to reduced saccade frequency, increased saccade latency, and impaired visual attention. For instance, in Parkinson's disease, dysfunction of the basal ganglia and its projections to the superior colliculus can result in hypometric saccades and increased latency due to impaired saccadic initiation and reduced dopaminergic input. Lesions in the basal ganglia, particularly the caudate nucleus, can disrupt the initiation and control of saccades. In Huntington's disease, slow and hypometric saccades are characteristic, stemming from impaired saccadic burst neuron activity. Additionally, saccadic intrusions and square-wave jerks are commonly observed in neurodegenerative disorders such as progressive supranuclear palsy and multiple system atrophy.[34,119,120]

Hypometric saccades are generally indicative of cerebellar dysfunction, specifically involving the cerebellar vermis, which plays a role in controlling the amplitude of saccades. Hypermetric saccades are often associated with dysfunction in the cerebellar output pathways, particularly involving the fastigial nucleus also known as the fastigial oculomotor region. The fastigial nucleus is involved in controlling saccadic accuracy, and lesions here can lead to overshooting (hypermetric saccades) due to impaired inhibitory control. For example, cerebellar ataxia often presents with hypermetric saccades due to disrupted cerebellar output to the brainstem saccadic generators. Slow saccades can also be a sign of brainstem lesions. Lesions affecting the PPRF can lead to horizontal saccade slowing, while lesions in the midbrain, particularly those involving the rostral interstitial nucleus of the medial longitudinal fasciculus (riMLF), can impair vertical saccades.[34,47,121]

Pursuit Movements

Pursuit movements are voluntary, smooth, and continuous eye movements, with their velocity and trajectory determined by the motion of the visual target.[77] These movements require a moving visual stimulus to initiate and can smoothly track speeds up to approximately 30 degrees per second; beyond this speed, pursuit movements often become interspersed with catch-up saccades to maintain foveation.[122] Interestingly, this threshold velocity correlates with the brain's processing capacity for high-acuity vision, as the visual cortex processes dynamic visual information at approximately 15 frames per second, and the fovea covers about 2 degrees of the visual field.[123 127]

Pursuit Circuitry

Neurons in the temporal eye field (TEF), located in the medial superior temporal (MST) and inferior parietal cortex, are involved in the initiation and guidance of smooth pursuit eye movements by computing the direction and velocity of a moving visual target.[128] These neurons project to the dorsolateral pontine nucleus (DLPN) and also send input to the FEF neurons, which, in turn, project to the DLPN. The FEF neurons initiate smooth pursuit based on input from the TEF neurons but do not directly guide the pursuit. The MST area is specifically responsible for processing motion signals and integrating them to generate appropriate smooth pursuit responses.[129,130]

The DLPN computes the direction and velocity of eye movement necessary to match the moving target's trajectory and sends excitatory projections that decussate and terminate in the contralateral cerebellum. Within the cerebellum, the flocculus and paraflocculus are particularly important for the modulation and maintenance of eye movements. These cerebellar regions send projections to the vestibular nuclei, which are involved in coordinating the antagonist muscle activities. The vestibular nuclei then send projections to the paraabducens nuclei.[130,131]

This double crossing in the horizontal smooth pursuit pathway results in eye movements in a direction ipsilateral to the cortical neurons. Through continuous feedback and adjustment, the cerebellum ensures that smooth pursuit remains accurate and stable, accommodating changes in target motion and maintaining foveation on moving objects. The MST area is involved in processing motion signals and integrating them to generate pursuit movements, while the cerebellar flocculus and paraflocculus are essential for maintaining the accuracy and stability of these movements (*Figure 6*).[98,102,131]

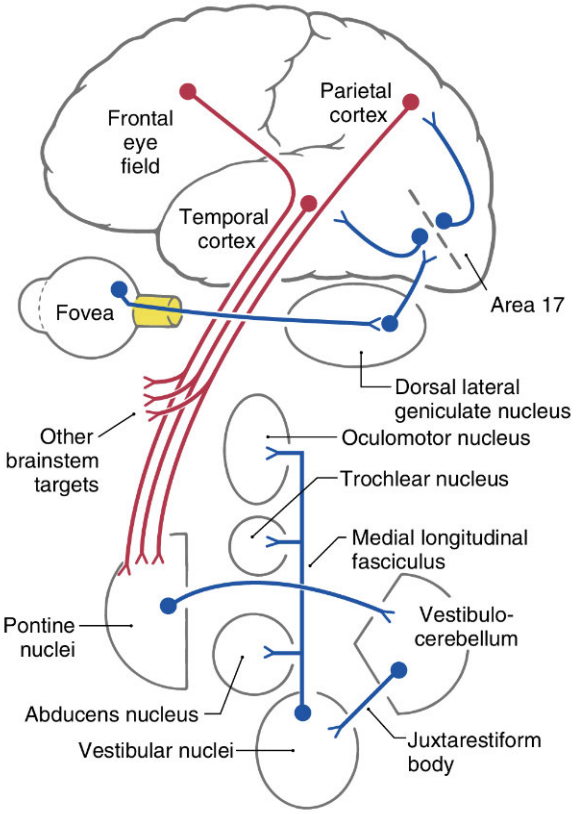


Figure 6: Smooth Pursuit pathway. [95]

Lesions in the FEF can lead to deficits in initiating smooth pursuit and impair predictive tracking of moving targets. Patients with FEF damage often struggle to initiate pursuit and may rely more on catch-up saccades due to impaired predictive control. This is particularly evident in patients with stroke or traumatic brain injury affecting the frontal lobes.[132] Lesions in the parietal cortex, especially the posterior parietal cortex, disrupt the integration of visual motion signals and attentional mechanisms necessary for smooth pursuit. This disruption can result in decreased pursuit gain and increased reliance on catch-up saccades to maintain foveation on moving targets. For example, patients with lesions in the right parietal cortex may exhibit ipsilateral pursuit deficits, such as reduced smooth pursuit velocity and accuracy when tracking rightward-moving targets.[133]

Basal ganglia involvement, particularly in diseases like Parkinson's disease, can affect smooth pursuit by disrupting the modulation of pursuit gain and the initiation of eye movements. Parkinsonian patients often exhibit hypometric smooth pursuit with frequent catch-up saccades, reflecting the diminished dopaminergic input to the pursuit pathways.[134] Lesions in the superior colliculus can also impair smooth pursuit by disrupting the integration of sensory input and motor output necessary for smooth tracking, resulting in both decreased pursuit gain and an increased number of catch-up saccades.[135,136]

Lesions involving the cerebellum, particularly the flocculus and paraflocculus, lead to a reduction in pursuit gain and an increased frequency of catch-up saccades. For instance, individuals with cerebellar degeneration or atrophy may present with severely impaired smooth pursuit, characterized by low gain and numerous catch-up saccades to compensate for the inability to smoothly follow moving objects. Disorders such as spinocerebellar ataxia often demonstrate these pursuit abnormalities.[1,22,45] Lesions in the dorsal pontine nuclei can disrupt the relay of visual motion information from the cortex to the cerebellum, resulting in pursuit impairments. Additionally, damage to the vestibular nuclei, which integrate signals for head and eye movements, can impair pursuit, especially during head movements, as seen in conditions like lateral medullary syndrome (Wallenberg's syndrome).[137,138]

Eye Tracking

Historically, various methods have been used to capture and analyze eye movements. Electrooculography (EOG) measures the electrical potential difference between the front and back of the eye using electrodes placed around the eyes. While effective for detecting large eye movements, EOG lacks the precision required for detailed gaze tracking. Scleral search coil systems, which involve placing a contact lens embedded with a wire coil on the eye, offer high precision by measuring induced voltage as the coil moves within a magnetic field. However, this method is invasive and uncomfortable for the patient.[139]

Modern eye trackers utilize cameras to capture images of the eye illuminated by infrared light (*Figure 7*). These systems are non-invasive, highly accurate, and well-suited for clinical applications. Unlike other methods, infrared eye trackers do not require direct contact with the eye and can achieve high precision in detecting eye movements, often with spatial resolutions below 0.1 degrees of visual angle. They typically employ infrared LEDs that provide consistent, non-intrusive illumination, invisible to the human eye (700-900 nm), and high-speed cameras capable of capturing images at rates ranging from 300 Hz to over 1200 Hz. This high capture rate is particularly advantageous for detecting subtle eye movement abnormalities.[140 142]

Figure 7: Tobii Pro Spectrum eye tracker.[143]

Visual stimuli are presented to the subject on a display, which can either be integrated with the eye tracker or connected to a separate monitor. A dedicated processing unit calculates the point of gaze from the captured images in real-time and transmits the data to a connected computer, where specialized software manages data acquisition. To determine the point of gaze, the camera captures reflected infrared light, producing images that emphasize the bright pupil and corneal reflection. The infrared light reflecting off the retina creates a bright pupil effect, similar to the red-eye effect in photography, enhancing the contrast between the pupil and the surrounding iris. Additionally, the infrared light reflects off the cornea, producing a glint or corneal reflection.[141]

The point of gaze is determined by analyzing the relative positions of the pupil and the corneal reflection. Advanced image processing algorithms, such as the Hough Transform, are employed to detect the circular shape of the pupil in the captured images. Techniques like edge detection and thresholding further isolate the pupil from the surrounding iris, enhancing detection accuracy. Geometric relationships between the eye's center, the corneal reflection, and the camera are then used to calculate the gaze point. Polynomial regression is applied to map the detected 3D geometry of the eye position to screen coordinates. Subsequently, post hoc data analysis software extracts and interprets information from the recorded eye movement data (*Figure 8*).[144 146]

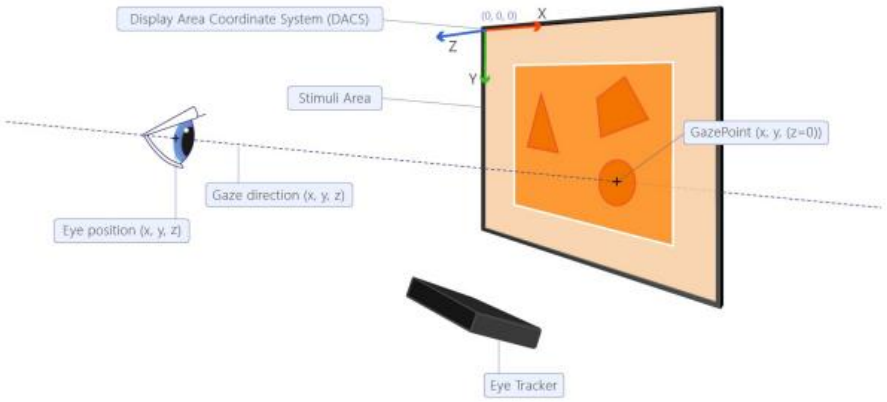


Figure 8: 3D coordinate system.[147]

Eye-Tracking Test

The eye-tracking test is designed to evaluate various aspects of eye movement through a systematic and controlled procedure. The patient is comfortably seated with a clear view of the screen, and their head is stabilized using an adjustable chinrest to minimize head movements. The eye tracker should be centrally positioned at an arm length distance from the patient's eyes to ensure measurement accuracy. Patients receive clear instructions about the procedure and its purpose, emphasizing the importance of remaining still, maintaining focus on the visual target, and following the stimuli with their eyes while avoiding head movements. Regular rest intervals may be provided to prevent eye strain and ensure sustained performance throughout the test.

The testing environment should be controlled for lighting and distractions to maintain consistent conditions for all patients. Proper screening for patient factors that might influence the results, such as visual acuity, cognitive status, fatigue, or medication use, is essential to ensure the reliability and validity of the test outcomes. Overall, the patient's ability to comprehend and adhere to the instructions, their physical capability to perform the test, the precise setup of the equipment, and the control of external variables are all critical components for obtaining reliable eye-tracking data.[8,142,148,149]

Calibration Process

During calibration, a series of targets are displayed sequentially at various points on the screen for a few seconds, and the patient is instructed to fixate on each target as it appears. The eye tracker collects data on the patient's eye position, creating a personalized model of the eyes' geometrical characteristics and the spatial relationship between the eyes and the screen. This process is repeated until acceptable accuracy is achieved. Calibration defines and adjusts the parameters necessary to transform the 2D image of the eye, captured by the tracker, into accurate 3D coordinates. These parameters account for the user's specific eye characteristics within the testing environment and can be influenced by individual anatomical variations, device positioning, and environmental conditions such as lighting. Proper calibration ensures that this transformation is precise, enabling the system to correctly and reliably map the 3D gaze coordinates.[142,150,151]

Visual Stimuli

During the eye-tracking test, a series of protocols are employed, each designed to evaluate specific aspects of eye movement. These protocols involve presenting visual stimuli on the display along with a task tailored to the purpose of the test and the aspect of eye movement being analyzed. For example, a moving dot across the screen with a task to follow it is used for the assessment of smooth pursuit.[123,140] Static stimuli include stationary targets, such as dots, which the patient is instructed to fixate on. These are useful for assessing fixation stability and identifying disorders related to fixational eye movements. Dynamic stimuli involve moving targets, such as a dot traveling across the screen in a predictable pattern, and are used to evaluate smooth pursuit. Complex stimuli require the integration of higher-order cognitive functions with ocular motor control, such as anti-saccades, where the patient must look in the opposite direction of a stimulus.[152,153]

Data Processing

The eye tracker continuously captures the patient's eye gaze data in response to stimuli with high spatial and temporal resolution. This data is streamed in real-time to a connected computer, where specialized software records the raw data and applies advanced algorithms for processing and analysis. The data processing workflow includes transformation, interpolation, denoising, and normalization.[142] Transformation algorithms convert the raw 3D geometric coordinates of eye positions, initially recorded in pixels, into units of interest such as millimeters or degrees. After transformation, interpolation techniques fill short periods of data loss to maintain continuity while preserving data integrity. These algorithms estimate the missing data points based on the surrounding data, commonly using linear interpolation.[30,145]

The raw data typically consists of time-series coordinates (x, y) representing gaze points over time. However, this data is often contaminated with noise and artifacts from various sources, including blinks, eye tracker hardware, environmental disturbances, and the intrinsic electronic digitization of analog input. Several techniques can be employed for denoising and smoothing the data, each with its own advantages and limitations. The choice of method depends on the study's objectives and the nature of the data. Effective denoising should differentiate genuine eye movement signals from noise while retaining the essential characteristics of the original signal, particularly edges and deflections that may correspond to true eye movement aberrations. Distortion of these features can lead to incorrect and potentially misleading results.[148,154,155]

Low-Pass Filter: This simple method is commonly used to remove high-frequency noise and interference, such as muscle contractions in EEG signals. However, it is not well-suited for eye-tracking because it removes high-frequency components that are not noise while retaining low-frequency noise, which is more common in eye-tracking data.[156]

Moving Average: This method replaces each data point with the average of its neighbors within a sliding window. The effectiveness depends on the window size in a manner in which a larger window may blur important transitions, while smaller windows may fail to filter out noise effectively. Additionally, it is also prone to smoothing noise, making it appear as genuine eye movement.[157]

Gaussian Filter: This method uses weighted averaging based on a Gaussian distribution. It offers better edge preservation than a simple moving average but may still significantly smooth out deflections if the data does not follow a Gaussian distribution, which is often the case with eye movement data.[158]

Moving Median: This method reduces noise by smoothing out spikes and irregularities, replacing each data point with the median value within a sliding window. It is reliable for addressing short-term fluctuations with a small window size but can significantly distort data with long-term fluctuations when a large window size is used.[159]

Wavelet Transform: An advanced computational method, wavelet transform decomposes the signal into different frequency components and applies thresholding to remove noise. It is a time-based derivative of Fourier transform and is highly effective for waveform signals, such as digital audio noise cancellation in headphones.[160]

$$W_\psi(s,u) = \int x(t)\psi\left(\frac{t-u}{s}\right) dt$$

$W_\psi(s,u)$ is the wavelet transform coefficient, $x(t)$ is the input signal, $\psi(t)$ is the mother wavelet, s is the scale parameter, u is the translation parameter, and ψ denotes the complex conjugate of the mother wavelet.

Savitzky-Golay Smoothing: This advanced computational method fits a polynomial to a window of data points and evaluates the polynomial at the central point. This process is repeated for each data point within a sliding window, resulting in a smooth approximation that preserves the integrity of the original signal. The polynomial coefficients are calculated using the least squares fitting method, making this technique particularly effective in preserving edges and deflections.[161]

$$y_i = \sum_{j=-k}^{k} c_j x_{i+j}$$

y_i is the smoothed data point, x_{i+j} are the raw data points, c_j are the coefficients derived from the least squares polynomial fitting, and k is the half-window size.

Following denoising, normalization is applied to correct systematic errors, ensuring that variations due to hardware inconsistencies or environmental factors are minimized. This process helps ensure that the recorded gaze points accurately represent the actual gaze points. These adjustments are based on the technical parameters of the eye tracker, such as its accuracy, precision, and degree of error.[162]

Data Analysis

Accurate calculation of biometric parameters relies on the precise detection of specific eye movement events. Event detection involves identifying fixations (where the eye remains relatively stationary) and saccades (rapid eye movements between fixations). This phase is particularly challenging due to physiological variations, noise, artifacts, and movements unrelated to the actual task. Common methods, such as velocity thresholds, are used to delineate these events.[19,142]

Following event detection, biometrics such as average fixation duration, spatial dispersion of fixations, saccade amplitude, peak velocity, and saccade duration are calculated. Additionally, visual representations of eye movement data, such as heat maps and fixation distributions, are generated to illustrate areas of high fixation concentration and variability in fixation points.[8,148]

Methods

All procedures involving human subjects were conducted in accordance with the ethical standards and guidelines of the institutional and national research committee, and with the 1964 Helsinki Declaration and its later amendments or comparable ethical standards. Prior to the commencement of the study, the research protocol was reviewed and approved by the University of KwaZulu-Natal Biomedical Research Ethics Committee (BREC/00006696/2024), the Provincial Health Research Committee of the KwaZulu-Natal Department of Health (NHRD Ref: KZ 202403 020), and Inkosi Albert Luthuli Central Hospital in Durban, South Africa (*Appendix 3*).

Experimental design

Participants

Participants were recruited through voluntary participation from the staff and patient population of the Department of Neurology at Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal, Nelson Mandela School of Medicine, Durban, South Africa, between June 24, 2024, and August 1, 2024. The healthy group comprised staff members from the Departments of Neurology and Neurophysiology, with study sessions conducted at their convenience during off-duty hours. The patient group included individuals from the Neurology outpatient clinic and inpatients admitted to the Neurology ward. For clinic patients, the study was conducted on the same day as their clinic visit, while for ward patients, it was conducted at a time convenient for them during their hospital stay.

Prospective participants received general information about the study, and those expressing interest underwent eligibility assessments, including a health and diagnostic review. Detailed medical histories were collected to identify potential confounding factors such as current medications, comorbidities, and primary diagnoses. Comorbidities were categorized using a predefined list, including: cardiovascular disease, diabetes mellitus, hypertension, immunocompromised conditions, kidney disease, hepatic disease, none, pulmonary disease, raised body mass index, psychiatric disorders, rheumatologic disease, and smoking. Ethnicity was also categorized using a predefined list: African, Asian, European, Middle Eastern, Mixed, Other, or South American.

Primary diagnoses were similarly categorized as follows: AIDP, atypical parkinsonism, autoimmune disorder, autonomic disorder, brainstem disorder, cerebellar disorder, channelopathy, CIDP, cognitive impairment, COVID-related disorder, cranial nerve disorder, dyssomnia, dystonia, encephalitis, encephalopathy, headache, hereditary neuropathy, HIV-associated disorder, HTLV-1–associated disorder, Huntington disease, hyperkinesia, idiopathic intracranial hypertension, intracranial hypotension, ischemic stroke, leukoencephalopathy,

meningitis, metabolic neuropathy, migraine, mononeuropathy, mononeuropathy multiplex, motor neuron diseases, movement disorder, MOGAD, multiple sclerosis, myelitis, myelopathy, neuromuscular disorder, neuropathic pain, NMOSD, normal, normal-pressure hydrocephalus, Parkinson disease, plexopathy, prion disease, sarcoidosis, seizure disorder, secondary malignancy, structural disorder, transient ischemic attack, tremor, and vascular disorders.

Candidate suitability was further assessed through the Mini-Mental Status Exam (MMSE). For the healthy group, visual acuity and fundoscopic examinations were not conducted, as participants were included based on known acceptable visual acuity. Similarly, in the patient group, these assessments were not repeated, as all neurology patients at the hospital routinely undergo visual acuity and fundoscopic examinations during their clinical evaluations, with results documented in their medical records. Patients with significant peripheral visual field deficits or enlarged blind spots were excluded.

The study's setup required a functional visual field extent of 44° horizontally and 22° vertically (normal visual field: 180° horizontal, 135° vertical). The blind spot is located approximately 12–15 degrees temporally and 2 degrees caudally.[163] Given the localization of the blind spot within the eye-tracking testing field, blind spot enlargement was considered a significant factor impacting the performance of the eye-tracking study. Consequently, blind spot assessment was a specific focus during participant evaluation.

Informed consent was obtained from individuals meeting the inclusion criteria (Appendix 2). The consent process involved providing participants with comprehensive information about the study's purpose, procedures, potential risks, and benefits, ensuring their full understanding and voluntary agreement to participate. Data handling procedures were meticulously designed to ensure the security and integrity of the collected information, with all data anonymized and securely stored to prevent unauthorized access.

Inclusion and Exclusion Criteria

Participants were divided into two groups: healthy subjects and patients with neurological diseases. Inclusion criteria included being 18 years of age or older, having normal or corrected-to-normal vision, the ability to understand and follow instructions, willingness and ability to provide informed consent, and the clinical and physical capability to complete the eye-tracking study. Exclusion criteria included a history of comorbid psychiatric conditions, a history of eye diseases or surgeries, a history of photosensitive migraine or epilepsy, a history of illicit substance abuse, current use of medications that may affect eye-tracking performance, and any other condition that could interfere with the eye-tracking testing.

Participants were seated (or positioned, if in a wheelchair) at a desk in front of the display, with their heads resting on an adjustable chinrest. A calibration procedure was conducted before each test, during which the chinrest height and the eye tracker were adjusted for both the participant's comfort and to align their primary line of sight with the center of the screen. Participants were offered breaks between protocol sessions, and if a break was taken, the calibration process was repeated. Participants could also request breaks at any point during the recording, with the recording process being restarted as needed. During each recording, the proctor monitored real-time gaze location on the host monitor using vendor-provided software (Tobii Pro Lab) to ensure compliance. The ambient luminance of the testing room was maintained at 60 ± 10 lux throughout the experiment, and participants were shielded from external distractions to maintain focus during the test sessions (*Figure 9*).

The calibration process was conducted using the default protocol provided by the Tobii Pro Eye Tracker Manager software. Through calibration, after verifying head position, a five-point calibration was performed, during which participants were instructed to fixate on calibration targets as they appeared on the screen. If necessary, the calibration process was repeated until satisfactory accuracy was achieved. Following calibration, the main diagnostic eye-tracking study protocol battery was conducted. The stimuli were presented using the vendor-provided software, Tobii Pro Lab. The entire eye-tracking test session lasted approximately 15 minutes.

A custom software application was developed by the principal investigator to automate the entire testing process (*Appendix 1*). This software was coded to streamline the testing procedure, data processing, artifact elimination, data analysis, and the qualitative and quantitative presentation of results. It manages participant information acquisition, including demographics, initiates the "Tobii Pro Eye Tracker Manager" for calibration, and subsequently launches "Tobii Pro Lab" for stimuli presentation and recording. Upon completion of all recordings, the software automatically exports the recorded data from Tobii Pro Lab, performs data processing and analysis, calculates biometrics, generates trajectory plots, and creates a PDF report, thereby eliminating the need for manual interactions or case-based adjustments. Additionally, the software stores the quantitative results in a built-in database (Data Bank) for future meta-analyses.

Figure 9: Eye-tracking test setup (subjects consented to publish these images).

Technical Parameters

The Tobii Pro Spectrum eye tracker used in this project offers a sampling frequency of 600 Hz. The device's accuracy is notable, with head-supported accuracy down to 0.15 degrees and precision maintained at 0.06 degrees RMS (Root Mean Square) under optimal conditions. The mean latency of the eye tracker is less than 2 milliseconds, with a gaze recovery time of less than 150 milliseconds. The eye tracker is equipped with a 23.8-inch IPS LED backlight screen (EIZO FlexScan EV2451) featuring a 16:9 aspect ratio and a resolution of 1920 × 1080 pixels, pre-mounted for stimuli presentation.

Given the operating distance of 55-70 cm from the screen, the maximum horizontal gaze angle is ±25 degrees from the center (50 degrees visual angle), and the maximum vertical gaze angle is ±14 degrees from the center (28 degrees visual angle). The eye tracker supports binocular eye tracking and provides raw, unprocessed gaze data. The Tobii Pro Spectrum is managed using the Tobii Pro Lab software platform, which facilitates stimuli presentation and data recording. Data export options include timestamps and 3D pixel coordinates of each eye gaze in Comma-Separated Values (CSV) or Microsoft® Excel (XLSX) file formats. The Tobii Pro Spectrum eye tracker complies with various international quality and safety standards.

The connected computer used for stimulus presentation, data recording, processing, and analysis was a Dell® Precision 3580 Mobile Workstation with a 15.6-inch screen, Core i7-1360P processor, 16GB RAM, 512GB SSD, and Nvidia RTX A500 graphics card, running Windows 11 Pro. Tobii Pro Lab software was utilized for presenting the visual stimuli and recording the raw eye movement data, which was saved in XLSX format. The DETS software application was developed using Microsoft® Visual Basic (Visual Studio .NET Version 16.9). Visual stimuli were created using Microsoft® PowerPoint (Microsoft 365 Version 2407).

Visual Stimuli

The design of the stimuli target features a small red dot (RGB 255,0,0) with sharp borders, enclosed within a larger white dot (RGB 255,255,255) with fading borders, all set against a black background (RGB 0,0,0). The diameter of the red center is set to 0.2 degrees of visual angle to ensure full foveal coverage (*Figure 10*). Initially, the target appears at 200% magnification and then gradually shrinks to its specified size within one second. This approach ensures that the target is clearly visible at first, reducing the risk of visual strain or difficulty in locating the target while minimizing noise. The specified small size ensures that the patient's eyes remain precisely focused on the target. Larger targets, which exceed the spatial resolution detection of the eye tracker, can cause the patient's gaze to shift within the target area, introducing noise and potentially hindering the accurate detection of abnormal eye movement intrusions.

The choice of a red dot is based on color perception, as the red wavelength primarily stimulates the L-cone photoreceptors (Long/Red), which are more numerous than other types of cones in the human fovea. [164] The sharp border of the red dot provides a clear and distinct target for the patient to focus on, minimizing fixation errors. The larger white dot with fading borders on a black background elicits a strong response from contrast-detecting rod photoreceptors. The gradient effect creates a soft transition that naturally guides the patient's gaze toward the red dot at the center, facilitating focus. The overall target design captures the patient's attention, maintains fixation without causing distraction or discomfort, minimizes fixation errors, and enhances the accuracy and reliability of the eye-tracking measurements.

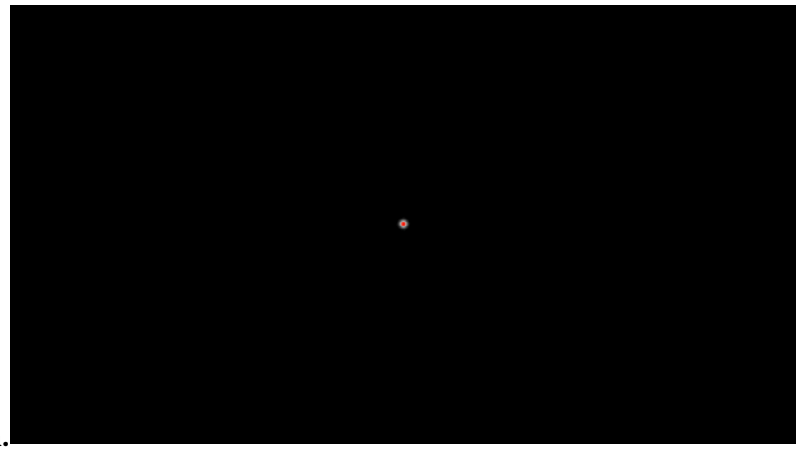
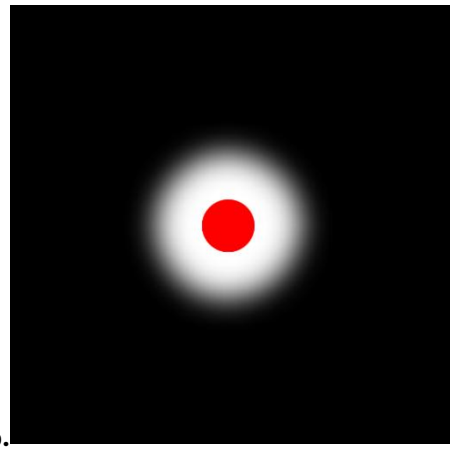
a.  **b.** 

Figure 10: a. Full screen stimuli target; b. Enlarged target.

Biometrics

Among quantifiable biometrics, the following metrics were defined to balance the ease and accuracy of automated calculation with the need for comprehensive, relevant, and informative measures in the context of clinical neurology. These biometrics provide insights into the functional integrity of the oculomotor network, thereby facilitating the understanding of various neurological disorders. By encompassing all aspects of oculomotor function, from voluntary and reflexive saccades to smooth pursuit and fixation stability, these measures ensure a thorough assessment of the neural circuits involved in eye movement control.

- **Accuracy:** The percentage to which the recorded gaze positions remain within tolerance limits of the actual stimuli target the patient is focusing on.
- **Variation:** The average degree and maximum deviation of data points from the actual stimuli target the patient is focusing on.
- **Congruency:** The average degree and percentage of the mean Euclidean distance between the right and left eyes' pairs of data points.

- **Saccadic Latency:** The average time delay (in milliseconds) between the appearance of a target and the initiation of a saccadic movement towards it.
- **Saccadic Gain:** The ratio of the amplitude of the saccade to the actual amplitude of the target from the initial gaze position. An ideal saccadic gain is close to 1, indicating that the eye movement reaches the target accurately.
- **Saccadic Velocity:** The average speed of eye movements during saccades (degrees per second). Although saccadic duration is also noted in literature, it is directly dependent on latency and velocity, making it a redundant measurement. Additionally, saccadic duration is test-dependent and not a constant measure for establishing normative data.
- **Pursuit Phase Shift:** The average phase (in degrees) of eye movements relative to the periodic target motion, analogous to latency in saccades.
- **Pursuit Gain:** The ratio of the amplitude of the eye position to the actual amplitude of the target during smooth pursuit. An optimal pursuit gain is close to 1, suggesting accurate tracking of the moving target.
- **Optokinetic Nystagmus (OKN) Gain:** The average difference between the amplitude of the eye positions at primary gaze and the actual primary gaze position.
- **OKN Variation:** The average degree of variation in the optokinetic response amplitude over time.
- **Prosaccades and Antisaccades Correct Rate:** The accuracy of saccadic eye movements towards (prosaccades) and away from (antisaccades) visual targets.
- **Prosaccades and Antisaccades Latency:** The average time delay (in milliseconds) between the appearance of a target and the initiation of eye movements.

Protocol Battery

A comprehensive battery of 17 test protocols was designed to evaluate various aspects of eye movement, with each protocol incorporating specific stimuli, relevant tasks, and biometric measurements. These protocols aimed to identify and analyze eye movement aberrations associated with or relevant to different neurological conditions. To ensure the highest quality of visual stimuli, each presentation was created using Microsoft® PowerPoint at a resolution of 1920 × 1080 pixels, perfectly aligning with the aspect ratio of the presenting screen to avoid any distortion. The stimuli were then converted to the Moving Picture Experts Group MPEG-4 Part 14 (MP4) video format. The conversion process was executed at a high frame rate of 170 frames per second, ensuring smooth and continuous playback.

Fixation

To assess Gaze Stabilization across the visual field. A static target is presented on the screen and the patient is instructed to fixate on the target for 12 seconds. Testing is conducted at the center (primary gaze) and at directional gaze angles.

- **Fixation Center:** The target is positioned at the center of the screen.
- **Fixation Right:** The target is positioned at +22 degrees visual angle from the center of screen in horizontal plane.
- **Fixation Left:** The target is positioned at -22 degrees visual angle from the center of screen in horizontal plane.
- **Fixation Up:** The target is positioned at +11 degrees visual angle from the center of screen in vertical plane.
- **Fixation Down:** The target is positioned at -11 degrees visual angle from the center of screen in vertical plane.

For each protocol, qualitative results are presented by two graphs (horizontal and vertical gaze angles over time) to visualize intrusions or instability. Quantitative results are presented by a table which includes accuracy, variation, and congruency biometrics for each eye.

Saccade

To assess task-specific saccadic eye movement (Gaze Shifting) for related neural pathways and identifying saccadic dysfunctions. Two targets appear alternately between two positions at the periphery of screen. Each remains static for 3 seconds before displacement, and the cycle is repeated 6 times. Testing is conducted across both horizontal and vertical directions.

- **Saccades Horizontal:** Targets are positioned at +22 degrees and -22 degrees visual angle from the center of screen in horizontal plane for a total displacement of 44 degrees.
- **Saccades Vertical:** Targets are positioned at +11 degrees and -11 degrees visual angle from the center of screen in vertical plane for a total displacement of 22 degrees.

For each protocol, qualitative results are presented by a graph plotting gaze angle over time to demonstrate saccadic abnormalities. Quantitative results are presented by a table which includes accuracy, variation, congruency, latency, gain, and velocity biometrics for each eye.

Pursuit

To assess smooth pursuit eye movement (Gaze Shifting) for related neural pathways and identifying dysfunctions. A target moves smoothly between two positions at the periphery of screen at a velocity of 5.5 degrees per second, completing 6 revolutions. The patient is instructed to follow the moving target with their eyes. Testing is conducted across both horizontal and vertical directions.

- **Pursuit Horizontal:** The target moves between +22 degrees and -22 degrees visual angle from the center of screen in horizontal plane for a total displacement of 44 degrees.
- **Pursuit Vertical:** The target moves between +11 degrees and -11 degrees visual angle from the center of screen in vertical plane for a total displacement of 22 degrees.

For each protocol, qualitative results are presented by a graph plotting gaze angle over time to demonstrate smooth pursuit aberrations. Quantitative results are presented by a table which includes accuracy, variation, congruency, phase shift, and gain biometrics for each eye.

OKN (Optokinetic Nystagmus)

To assess optokinetic reflex function (gaze stabilization) for related neural pathways and to identify dysfunctions. A series of targets, spaced 5 degrees of visual angle apart, move across the screen at a velocity of 5.5 degrees per second. This cycle is repeated 20 times. The patient is instructed to count the number of targets that appear at the center of the screen. Testing is conducted in both horizontal and vertical directions, with movement toward both the right and left.

- **OKN Right:** Targets move horizontally to the right.
- **OKN Left:** Targets move horizontally to the left.
- **OKN Up:** Targets move vertically upward.
- **OKN Down:** Targets move vertically downward.

For each protocol, qualitative results are presented by a graph plotting gaze angle over time to demonstrate OKN abnormalities. Quantitative results are presented by a table which includes gain and variation biometrics for each eye.

ProSaccades

To assess cognitive function, including attention, executive function, and response inhibition. Three targets are displayed on the screen. The patient is instructed to fixate on the center target as the primary focus. When one of the peripheral targets begins to blink, the patient shifts their gaze to the blinking target until it stops blinking. This cycle is repeated six times in a non-alternating fashion to resemble randomness and eliminate the anticipation confounder. Testing is conducted across both horizontal and vertical directions.

- **ProSaccades Horizontal:** Targets are aligned horizontally.
- **ProSaccades Vertical:** Targets are aligned vertically.

For each protocol, qualitative results are presented by a graph plotting gaze angle over time. Quantitative results are presented by a table which includes error rate and latency biometrics for each eye.

AntiSaccades

To further assess cognitive function, including attention, executive function, and response inhibition. Three targets are displayed on the screen. The patient is instructed to fixate on the center target as the primary focus. When one of the peripheral targets begins to blink, the patient shifts their gaze to the opposite static target until the blinking stops. This cycle is repeated six times in a non-alternating fashion to resemble randomness and eliminate the anticipation confounder. Testing is conducted across both horizontal and vertical directions.

- **AntiSaccades Horizontal:** Targets are aligned horizontally.
- **AntiSaccades Vertical:** Targets are aligned vertically.

For each protocol, qualitative results are presented by a graph plotting gaze angle over time. Quantitative results are presented by a table which includes error rate and latency biometrics for each eye.

Data Analysis

Compared to popular programming languages such as Python or R, VBA is limited in its availability of advanced mathematical and processing libraries and functions, necessitating the development of many components from scratch. Despite these limitations, VBA was selected for its strong integration with Excel and the need to compare automated analysis with manual processing for accuracy assessment, and our team's familiarity and expertise with VBA allowed for rapid development and implementation of the necessary code, leveraging existing skills to expedite the research process.

Data Processing

Importing & Segmentation

The initial step involves identifying different test protocols and importing them into separate files based on protocols in a directory named by participant number and date of study. Next, the recorded parameters in each file are sorted and reordered for further processing.

Transformation

For each eye, the horizontal (x) and vertical (y) gaze points of the raw samples (3D pixel coordinate geometry) were converted to spherical angles in degrees, with the patient's eye as the center and the patient's distance to the display as the radius. We employed the arctangent theorem for each data point in time to ensure the gaze angle calculation remains precise, accounting for any head movement that may alter the viewing distance. The screen's center was assigned as 0 degrees in both horizontal and vertical angles, with angles to the right or up defined as positive and angles to the left or down as negative.

$$\alpha^\circ = \frac{180}{\pi} \arctan\left(\frac{\text{hypotenuse}}{\text{adjacent}}\right)$$

Interpolation

We developed an advanced method to estimate and impute missing values by leveraging the time-series properties of the data. This approach dynamically incorporates the slope of the known data points preceding and following the gap, thereby capturing the underlying dynamics and stochastic properties of eye movements. By considering these temporal relationships, our method offers superior accuracy compared to commonly used imputation techniques.

```
Private Function FillEmptyData(ByRef fillDataArr As Variant) As Variant

    On Error GoTo ErrorHandler
    Dim i As Long, j As Long, k As Long, m As Long, lastRow As Long, lastCol As Integer
    Dim afterIndex As Long, beforeIndex As Long
    Dim afterValue As Double, beforeValue As Double, Slope As Double

    lastRow = UBound(fillDataArr, 1)
    lastCol = UBound(fillDataArr, 2)

    For i = 1 To lastCol
        For j = 1 To lastRow
            If IsEmpty(fillDataArr(j, i)) Then
                k = j
                Do While IsEmpty(fillDataArr(k, i)) And k < lastRow
                    k = k + 1
                Loop

                If k > lastRow Then
                    afterValue = 0
                    afterIndex = lastRow
                Else
                    afterValue = fillDataArr(k, i)
                    afterIndex = k
                End If

                k = j
                Do While IsEmpty(fillDataArr(k, i)) And k > 1
                    k = k - 1
                Loop

                If k < 1 Then
                    beforeValue = 0
                    beforeIndex = 1
                Else
                    beforeValue = fillDataArr(k, i)
                    beforeIndex = k
                End If

                If afterIndex - beforeIndex <> 0 Then
                    Slope = (afterValue - beforeValue) / (afterIndex - beforeIndex)
                    For m = beforeIndex + 1 To afterIndex - 1
                        fillDataArr(m, i) = Slope * (m - beforeIndex) + beforeValue
                    Next m
                End If
            End If
        Next j
    Next i

    FillEmptyData = fillDataArr

    Exit Function

ErrorHandler:
    Call ErrorHandler

End Function
```

De-noising

An optimized version of the Savitzky-Golay filter, the Dynamic Savitzky-Golay Filter, has recently been applied to fMRI time series data. This method dynamically adjusts the polynomial degree based on the signal's changing nature. The polynomial degree and window size determine how well the filter fits the data within the window. A higher-degree polynomial can model more complex variations in the data, closely following the underlying trend, especially in cases of high variability. However, higher-degree polynomials can also overfit noise and reduce the amplitude of sudden deflections. Conversely, lower-degree polynomials may underfit the signal, failing to capture finer details in the data.[165]

Our novel approach to the Dynamic Savitzky-Golay filter uses a dynamic window size which is adjusted through a loop based on the local variance and an adjusted threshold condition. This approach is particularly advantageous for processing signals with varying characteristics, such as eye-tracking data with different types of eye movements. With this method, sudden deflections in eye movement are not treated as noise, allowing for better noise reduction while preserving eye movement aberrations, and their original physical metrics such as amplitude.

The dynamic window size starts at the predefined maximum window size and is reduced iteratively if the variance of the regressed data exceeds the dynamic threshold. The sampling frequency determines the max window size, while the tolerance and threshold values, are based on the technical parameters of the eye-tracker machine. Adjustment of threshold is based on the size of the dynamic window, and controls the precision of the noise reduction algorithms.

```
Private Function Regressize(ByRef sgDataArr As Variant, ByVal maxWinSize As Integer, _
ByVal minWinSize As Integer, ByVal threshold As Double) As Variant

    On Error GoTo ErrorHandler
    Dim i As Long, n As Long, j As Long, dyWinSize As Integer, edgeAvg As Double, _
        dyWinReg As Variant
    n = UBound(sgDataArr)

    edgeAvg = CalcAverage(GetWinArr(sgDataArr, (maxWinSize \ 4) + 1, maxWinSize \ 2))
    For i = 1 To maxWinSize * 2 \ 3
        sgDataArr(i) = edgeAvg
    Next i

    edgeAvg = CalcAverage(GetWinArr(sgDataArr, n - (maxWinSize \ 4), maxWinSize \ 2))
    For i = n - maxWinSize * 2 \ 3 To n
        sgDataArr(i) = edgeAvg
    Next i

    For i = maxWinSize \ 2 + 1 To n - maxWinSize \ 2
        j = 0
        dyWinSize = maxWinSize
        If dyWinSize Mod 2 = 0 Then dyWinSize = dyWinSize + 1
        Do While dyWinSize >= minWinSize
            j = j + 1
            dyWinReg = CalcRegress(GetWinArr(sgDataArr, i, dyWinSize))
            If dyWinReg(1) > 10 ^ (threshold + j) Then
                dyWinSize = dyWinSize \ 2
```

```
                If dyWinSize Mod 2 = 0 Then dyWinSize = dyWinSize + 1
            Else
                sgDataArr(i) = dyWinReg(2)
                Exit Do
            End If
        Loop
    Next i

    Regressize = sgDataArr
    Exit Function

ErrorHandler:
    Call ErrorHandler

End Function

Public Function CalcRegress(ByRef dyWinArr As Variant) As Variant

    On Error GoTo ErrorHandler
    Dim i As Long, n As Long
    Dim xVal As Double, yVal As Double, xSum As Double, ySum As Double, xySum As Double, _
        xxSum As Double, Slope As Double, intercept As Double, yPred As Double, _
        varSum As Double, regVals() As Double
    ReDim regVals(1 To 2)

    n = UBound(dyWinArr)
    For i = 1 To n
        xVal = i
        yVal = dyWinArr(i)
        xSum = xSum + xVal
        ySum = ySum + yVal
        xySum = xySum + xVal * yVal
        xxSum = xxSum + xVal ^ 2
    Next i

    If n * xxSum - xSum ^ 2 = 0 Then
        Slope = Settings.AppSettings("Max Horizontal Angle")
    Else
        Slope = (n * xySum - xSum * ySum) / (n * xxSum - xSum ^ 2)
    End If

    intercept = (ySum - Slope * xSum) / n
    xVal = n - n \ 2
    regVals(2) = (Slope * xVal) + intercept

    For i = 1 To n
        xVal = i
        yVal = dyWinArr(i)
        yPred = (Slope * xVal) + intercept
        varSum = varSum + (yVal - yPred) ^ 2
    Next i

    regVals(1) = varSum / n

    CalcRegress = regVals
    Exit Function

ErrorHandler:
    Call ErrorHandler
End Function
```

Blink Artifact Removal

Potential blink artifacts are detected by iterating through the data and calculating slope and average values for each data point within the surrounding data points in a specified window. A potential blink point is identified if the slope at that point is zero and the slopes before and after the point change signs, indicating a discontinuity. The detected potential blink points undergo multiple rigorous validation steps, including slope, amplitude, area under the curve, and interval thresholding to avoid false positives. Once validated, the data within the blink intervals is replaced by interpolation of the data points before and after the blink. This method ensures a smooth and continuous transition, effectively removing the blink artifact while preserving the integrity of the eye-tracking data.

```
Private Function DeBlink(ByRef blDataArr As Variant, ByVal blWinSize As Integer, _
ByVal interval As Integer, ByVal threshold As Double, ByVal tolerance As Double) As Variant

    On Error GoTo ErrorHandler
    Dim i As Long, j As Long, k As Long, m As Long, n As Long, p As Long, foundBlink As Boolean
    Dim tempSlope As Double, tempAvg As Double, tempVal As Double, slopeArr() As Double, _
avgArr() As Double

    ReDim slopeArr(1 To UBound(blDataArr))
    ReDim avgArr(1 To UBound(blDataArr))

    For i = blWinSize * 2 + 1 To UBound(blDataArr) - blWinSize * 2
        tempSlope = CalcSlope(GetWinArr(blDataArr, i - blWinSize, blWinSize * 2))
        tempAvg = CalcAverage(GetWinArr(blDataArr, i - blWinSize, blWinSize * 2))

        If Abs(tempSlope) > threshold And Abs(tempAvg) > tolerance Then
            slopeArr(i) = tempSlope
            avgArr(i) = tempAvg
        Else
            slopeArr(i) = 0
            avgArr(i) = tempAvg
        End If
    Next i

    k = 0
    ReDim blinkPoint(1 To 100)

    For i = blWinSize * 20 + 1 To UBound(blDataArr) - blWinSize * 20
        If slopeArr(i) = 0 Then
            foundBlink = False
            For j = 1 To blWinSize * 20
                If slopeArr(i - j) * slopeArr(i + j) < 0 And _
Abs(avgArr(i - j) - avgArr(i + j)) < tolerance Then
                    For m = j To blWinSize * 20
                        If slopeArr(i - m) * slopeArr(i + m) = 0 Then
                            For p = m To blWinSize * 20
                                If slopeArr(i - p) = 0 And slopeArr(i + p) = 0 And _
Abs(avgArr(i - p) - avgArr(i + p)) < tolerance Then
                                    foundBlink = True
                                Else
                                    foundBlink = False
                                End If
                            Next p
                            If foundBlink = True Then Exit For
```

```
                    Next m
                    If foundBlink = True Then Exit For
                End If
            Next j

            If foundBlink = True Then
                If k < 1 Then
                    k = k + 1
                    blinkPoint(k) = i
                Else
                    If i > blinkPoint(k) + interval Then
                        k = k + 1
                        blinkPoint(k) = i
                    End If
                End If
            End If
        End If
    Next i

    If Not k = 0 Then
        For i = 1 To k
            tempVal = (blDataArr(blinkPoint(i) - blWinSize * 20) + _
blDataArr(blinkPoint(i) + blWinSize * 20)) / 2
            For j = blinkPoint(i) - blWinSize * 20 To blinkPoint(i) + blWinSize * 20
                blDataArr(j) = tempVal
            Next j
        Next i
    End If

    DeBlink = blDataArr

    Exit Function

ErrorHandler:
    Call ErrorHandler

End Function
```

Normalization

The normalization process ensures accuracy and consistency across all data points and protocols by leveraging the technical parameters of the eye-tracker along with central gaze values obtained from the initial protocol. A sigmoid function determines a weight factor based on the difference between each paired data point for the right and left eye, and the average value of data points within a specified window. This weight factor is then used to normalize the data, correcting any initial offsets in the gaze data and aligning all data points to a common baseline to account for inherent biases or errors in the test setup.

$$\omega = \frac{1}{1 + e^{-\beta(d-\tau)}}$$

ω is the weight factor, β is a parameter that controls the steepness of the sigmoid curve, d is the difference between the data point and the average value, τ is a threshold difference.

Event Detection

The calculation of biometrics first involves the precise identification of specific eye movement events. Since all video stimuli are precisely time-locked to the tasks and related events, a pilot time-series dataset representing the actual stimulus movement is added to the data set for event detection processes.

Saccadic Start and End Points Identification

Potential saccadic events surrounding the actual stimulus event of interest are identified by iterating through the data points and assigning them to a window. The distance between each data point in this window and the expected maximum deflection is calculated using the Euclidean distance formula. The data point with the least distance is marked as the start or end of the saccade accordingly. Multiple conditions are implemented to prevent noise or aberrations from being incorrectly marked as event points.

Baseline or Expected Trajectory Cross Points Identification

The same principle is applied to identify potential cross points. For each candidate cross point within the interval encompassing the actual stimulus event, the local minimum point is verified by comparing it with neighboring points. If the absolute eye position matches the baseline value and is a local minimum, the point is considered a cross point. After identifying cross points, interval validation is performed to ensure that detected points are sufficiently spaced apart, avoiding false positives due to noise or aberrant eye movements. The result is a refined list of cross points that accurately mark where the eye movements cross the baseline.

Peak (Maxima) and Trough (Minima) Points Identification

This process employs an approach similar to calculating the area under the curve. If the sum of the eye positions in a specified sliding window exceeds that of the preceding and succeeding intervals, it is considered a peak. Similarly, if the sum of the eye positions in a specified sliding window is the minimum, it is considered a trough. To avoid false positives, the function verifies that each detected point remains the highest or lowest within its interval by comparing it with neighboring points within a larger interval, ensuring it is the most significant peak or trough.

Biometrics Calculations

Accuracy

By iterating through each data point, the difference between the gaze point and the actual target is calculated. If this difference exceeds a predefined tolerance (set by technical parameters), it is counted as an error. The accuracy percentage is then derived by subtracting the proportion of errors from 1 and multiplying by 100.

$$Accuracy\ \% = \left(1 - \frac{number\ of\ errors}{n}\right) \times\ 100$$

Variation

The average and the maximum differences between each gaze point and the actual target.

$$Variation\ ^\circ = \frac{\sum \sqrt{(gaze\ point - target)^2}}{n}$$

Congruency

The average Euclidean distances between the paired gaze points of the right and left eyes.

$$Congruency\ ^\circ = \frac{\sum \sqrt{\left(gaze\ point_{right} - gaze\ point_{left}\right)^2}}{n}$$

Latency

The average time difference between the identified saccadic initiation events and the onset of the target appearance.

$$Latency\ ms = \frac{\sum(time\ of\ saccade - time\ of\ stimulus)}{number\ of\ saccades}$$

Gain

The ratio of the sum of the actual gaze amplitude to the sum of the target amplitude.

$$Gain\ ^\circ = \frac{\sum(gaze\ amplitude)}{\sum(target\ amplitude)}$$

Velocity

The average change in gaze positions between initiation and termination of events over the duration of intervals.

$$Velocity\ ^\circ/s = \frac{\Delta\ gaze\ position}{\Delta\ time}$$

Phase Shift

The average phase differences between the gaze cross points and the target cross points.

$$Phase\ Shift\ ^\circ = \left(\frac{180}{cycle\ duration}\right) \times (time\ of\ gaze - time\ of\ target)$$

Correct Rate

The total number of correct responses based on predefined criteria, such as moving the gaze to the target within a specified time, direction, and magnitude.

Statistical Analysis

All statistical analyses were performed using R version 4.4.1 (The R Project for Statistical Computing, Vienna, Austria). A significance level of 0.05 was considered statistically significant. Given the exploratory nature of this pilot study, the primary statistical objectives were to validate the DETS protocols and establish normative biometric values with high precision. The sample size was determined based on preliminary estimates, and post hoc analyses to ensure adequacy. Specifically, the narrow 99% confidence intervals across all analyzed biometrics confirmed that the sample size was sufficient to yield reliable and robust estimates, indicating that the collected data are representative of the population.

The normality of data distribution was assessed using the Kolmogorov-Smirnov test to guide the selection of appropriate statistical methods for subsequent analyses. Welch's t-test was chosen over the traditional Student's t-test for comparing DETS performance, given the potential for unequal variances between groups, thereby providing a more robust comparison. Cohen's d was utilized to measure effect size, quantifying the magnitude of observed differences between groups. For each eye-tracking biometric, the lower and/or upper bounds of the 99% confidence intervals were calculated to establish normative values, offering a reference range for distinguishing between normal and pathological findings with high confidence. Additionally, the DETS software generated qualitative visualizations of eye movement trajectories, which were evaluated in conjunction with the statistical data.

Results

The primary aim of this project was to develop the Diagnostic Eye-Tracking Study (DETS) software (*Appendix 1*) to automate comprehensive testing and analysis relevant to clinical neurology, and to validate its accuracy, efficacy, and performance through a pilot study.

The main user interface of DETS includes several fields for patient information: a unique institutional ID number, last name, first name, date of birth, and study date. Additionally, it features sections for biological sex (with toggle buttons for male and female), ethnicity (selectable from a dropdown menu), Mini-Mental State Examination (MMSE) score, diagnosis (selectable from a dropdown menu), and comorbidities (multi-select checkboxes), as well as a text box for additional notes (*Figure 11*). The interface also includes several action buttons for tasks such as creating a new record, importing a recording, analyzing data, reviewing reports, re-analyzing data, and accessing the data bank.

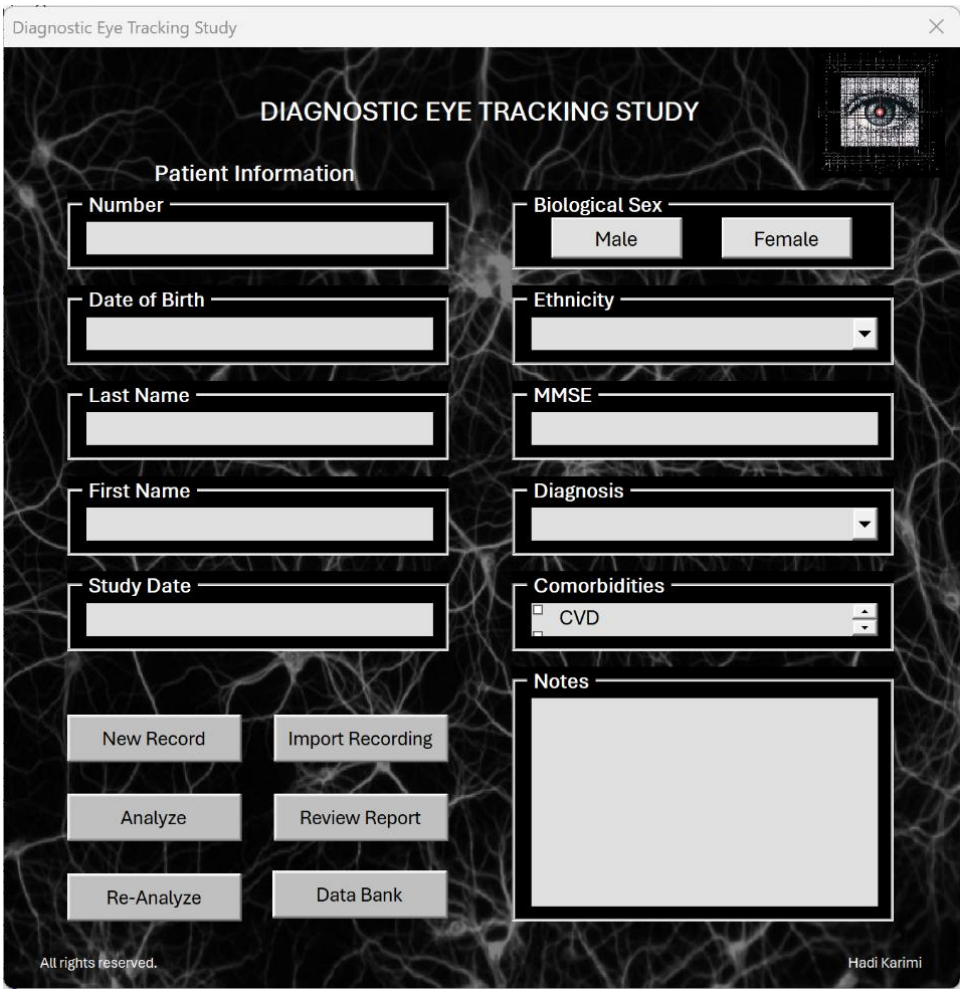


Figure 11: DETS main user interface.

Participant Demographics

The pilot study comprised 22 participants (*Table 3*), including 16 healthy individuals and 6 patients with neurological disorders. Within the healthy cohort, the mean age was 37 years (range: 22-65), with 7 males (44%) having a mean age of 40 years (range: 23-64) and 9 females (56%) with a mean age of 35 years (range: 22-65). The ethnic distribution of the healthy group included 9 individuals (56%) of African descent, 4 (25%) Asian, 1 (6%) Middle Eastern, and 1 (6%) of European descent. All healthy participants had a Mini-Mental State Examination (MMSE) score of 30 and Visual Acuity better than 0.2 LogMAR. Two individuals had comorbid conditions (Hypertension, Diabetes Mellitus).

Within the patient cohort, the mean age was 46 years (range: 28-68), with 3 males (50%) having a mean age of 53 years (range: 28-68) and 3 females (50%) with a mean age of 39 years (range: 33-43). The ethnic distribution of the patient group included 3 individuals (50%) of African descent, 2 individuals (33%) of Asian, and one European (16%). The neurological disorders represented within the patient cohort included Friedreich's ataxia, Multiple Sclerosis, Myasthenia Gravis, Progressive Supranuclear Palsy, Posterior Circulation Stroke, and Cerebellar degeneration. Mini-Mental State Examination (MMSE) score ranged from 26 to 30, and they all had Visual Acuity better than 0.2 LogMAR. Comorbid conditions were present in three patients: one patient had Hypertension, and three were immunocompromised. The duration of the neurological disorder varied among the patients, with a mean disease duration of 5 years (range: 2 month - 9 years).

Table 3: Demographics of the Pilot Study

Characteristic	Healthy Cohort	Patient Cohort
Number	16	6
Age Mean (range)	37 (22-65)	46 (28-68)
Male	7 (44%)	3 (50%)
Female	9 (56%)	3 (50%)
Black	9 (56%)	3 (60%)
Indian	5 (31%)	2 (33%)
Middle Eastern	1 (6%)	0
White	1 (6%)	1 (16%)
MMSE Mean (range)	30 (30)	29 (26-30)
Visual Acuity (LogMAR)	<0.2	<0.2
Comorbidities	2 (13%)	3 (50%)
Hypertension	1	1
Diabetes Mellitus	1	0
Immunocompromised	0	3
Neurological Disorders	-	Friedreich's ataxia, Multiple Sclerosis, Myasthenia Gravis, Progressive Supranuclear Palsy, Posterior Circulation Stroke, Cerebellar degeneration
Disease Duration Mean (range)	-	5 years (2 month-9 years)

The statistical analysis comparing the healthy cohort and the patient cohort revealed no significant differences across the variables assessed. The chi-square test for gender distribution showed no significant difference between the groups, with a chi2-statistic of 0.015 and a p-value of 0.903. Similarly, the ethnic distribution between the cohorts did not differ significantly, with a chi2-statistic of 0.889 and a p-value of 0.828. Overall, we can conclude that the demographic of the healthy cohort and the patient cohort in this study are not significantly different in terms of biological sex distribution, ethnic distribution, and the presence of comorbid conditions. This lack of significant differences suggests that the two groups are relatively well-matched across these key variables. However, it's important to note that the small sample size, especially in the patient group, may limit the power of these statistical tests to detect differences, especially for age distribution, and these results should be interpreted with caution.

DETS Validation

The validity and reliability of the Diagnostic Eye Tracking System (DETS) were assessed through a multifaceted approach, comparing its accuracy, effectiveness, and data integrity against manual measurements and/or unprocessed raw data. Cross-validation techniques were applied to ensure the generalizability of the results, and inter-rater reliability was assessed to confirm consistency in manual measurements.

Data Loss and Artifact Rate

Data loss was calculated as the percentage of total data points that were unavailable for analysis due to technical failures (signal loss during data collection). The overall data loss across all participants was found to be 1.6%. This was further stratified by participant group, revealing a data loss of 1.3% in healthy participants and 2.1% in patients. To determine if the difference in data loss between the healthy participants and patients is statistically significant, Welch’s t-test was performed due to the potential inequality of variances between groups. The test resulted in a p-value of 0.13, indicating that the difference in data loss between the healthy participants and patients is not statistically significant.

The artifact rate was assessed as the percentage of data points contaminated by clinically non-relevant signals, such as blinks or signal noise, in the unprocessed data. The overall artifact rate was 11.7%, with a rate of 11.5% in healthy participants and 12.1% in patients. A Welch’s t-test for artifact rates yielded a p-value of 0.81, suggesting no significant difference between the two groups. After post-processing, the artifact rate was significantly reduced to 4.2% overall, with 3.7% in healthy participants and 4.9% in patients, with no statistical significance between the two groups (p-value 0.64).

Signal-to-Noise Ratio Improvement

The Signal-to-Noise Ratio (SNR) was quantified to assess the effectiveness of the denoising process. SNR was measured both before and after the application of the denoising algorithm. The SNR of the raw signal was calculated by comparing the power of the signal component to the power of the noise component. Results demonstrated a substantial improvement in SNR, with an overall increase of approximately 20 dB. The improvement was slightly more pronounced in healthy participants, with a 21 dB increase, compared to a 21 dB increase in patients. The effect size for healthy group, calculated using Cohen's d, was approximately 5.63, indicating a large effect and the significant impact of the denoising algorithm. A Welch's t-test comparing the SNR improvement between the healthy participants and patients yielded a p-value of 0.52, indicating no statistically significant difference between the groups (*Figure 12*).

Distorted Signal Rate and Root Mean Square Error

The distorted signal rate was quantified as the percentage of the total clinically relevant signal that was considered altered after processing with DETS. These distortions could arise from various sources, such as over-smoothing, non-blink signal removal, or reduction in amplitude. The overall distorted signal rate across all participants was 0.01%, with a similar distribution between healthy participants and patients (0.0096% and 0.0098%, respectively). Moreover, the Root Mean Square Error (RMSE) was employed as a metric to quantify the accuracy of the signal reconstruction post-processing. RMSE was calculated by comparing the original raw signal with the denoised signal, providing an average measure of the magnitude of error between these two signals. The overall RMSE across all participants was 0.01. When stratified by group, the RMSE was 0.01 for healthy participants and 0.03 for patients.

Statistical Validation

A Kolmogorov-Smirnov test was conducted to compare the distributions of raw data with the results of DETS to evaluate if the data processing introduced any significant alterations in the data distribution. The result indicated no significant difference between the distributions for all results, supporting the integrity of the data after processing (overall $p = 0.16$, healthy average $p = 0.19$, patient average $p = 0.11$).

Qualitative Feedback

Qualitative feedback was gathered from participants post-testing, with no participants reporting difficulties in locating the target, losing concentration, or experiencing strain during the tests.

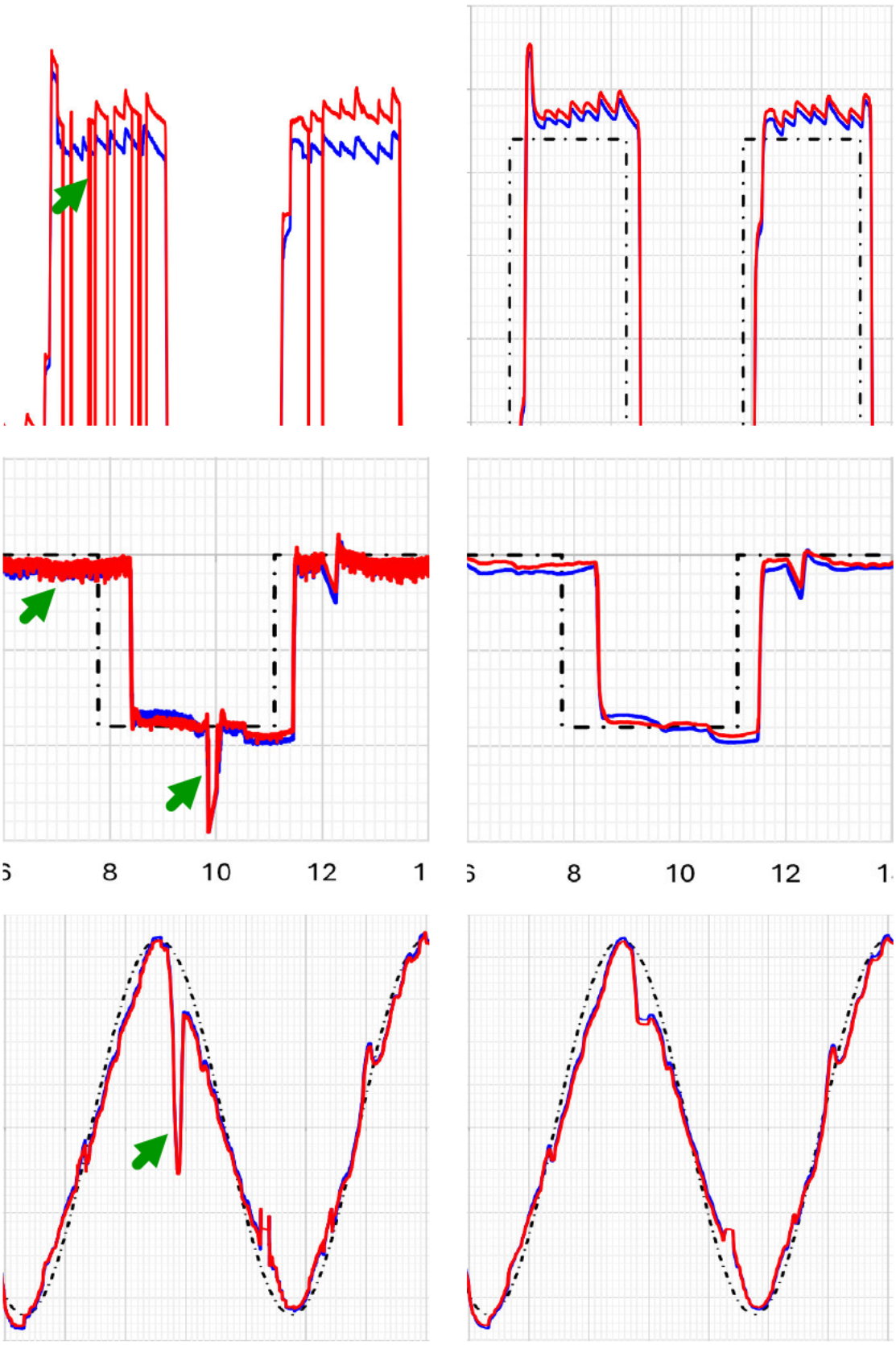


Figure 12: Time-series visual inspection analysis. Left side: pre-processed, Right side: post-processed. Samples highlight the reduction in noise and artifacts removal such as blink (green arrows) while preserving stochastic eye movements integrity. The pilot black dotted line represents actual target movement, and the red and blue line represents subject's gaze (red: right eye, blue: left eye).

Protocols Reports

The DETS report provides quantitative biometrics results in the form of a table at the bottom of each protocol report. Complement to the quantitative results, it also offers a qualitative result through time-series trajectory plots (*Figure 13*). These graphs show the eye movement trajectory in both horizontal and vertical planes to capture the full spectrum of ocular motor performance for a visual inspection. Table 3 provides a statistical overview of the eye-tracking biometrics across various protocols for the healthy participants. The defined 99% confidence intervals provide benchmarks (Normative Values) for identifying deviations from normal range, which could be indicative of underlying neurological conditions (*Table 4*).

Fixation

The analysis of fixation protocols across different directions (center, right, left, up, and down) consistently demonstrated high accuracy levels, with mean values exceeding 98% across all directions. Central fixation exhibited the highest mean accuracy of 98.91% (SD = 1.75), with minimal variability, as indicated by the relatively low standard deviation. This suggests that healthy participants maintain their gaze centrally with high precision and consistency. The 99% confidence interval (CI) lower bound of 97.32% establishes a threshold below which accuracy may be indicative of abnormal fixation performance.

Right and left fixation accuracy was slightly lower, with mean values of 98.57% (SD = 1.96) and 98.71% (SD = 2.78), respectively. The 99% CI lower bounds of 96.78% for right fixation and 96.18% for left fixation serve as benchmarks for potential clinical significance. The increased standard deviation for these directions suggests greater variability in maintaining gaze compared to central fixation. Upward and downward fixations showed mean accuracies of 98.59% (SD = 2.26) and 98.90% (SD = 2.52), with corresponding 99% CI lower bounds of 96.53% and 96.60%.

In terms of fixation variation, central fixation had the lowest mean variation at 0.13 degrees (SD = 0.06), with an upper bound of 0.19 degrees in the 99% CI. Variations in right and left fixations were slightly higher, with means of 0.17 degrees (SD = 0.13) and 0.15 degrees (SD = 0.08), respectively. The upper bounds of 0.29 degrees for right fixation and 0.23 degrees for left fixation indicate that deviations beyond these values may signify instability or intrusions. Upward and downward fixations exhibited slightly greater variations, with means of 0.19 degrees (SD = 0.16) for up and 0.14 degrees (SD = 0.08) for down, with upper bounds of 0.33 degrees and 0.21 degrees in the 99% CI, respectively.

Congruency biometrics revealed consistent performance, with central fixation demonstrating the most stable congruency at 0.25 (SD = 0.13) and an upper bound of 0.37 in the 99% CI. Right fixation had a mean congruency of 0.35 (SD = 0.11) and an upper bound of 0.45, suggesting slightly more variability in eye alignment. Left fixation showed the highest mean congruency at 0.41 (SD = 0.18) and the broadest variability, with an upper bound of 0.57 in the 99% CI. Upward and downward

fixations exhibited congruency means of 0.37 (SD = 0.15) and 0.43 (SD = 0.13), with 99% CI upper bounds of 0.50 and 0.55, respectively.

The fixation protocols' graphical illustrations confirmed participants' ability to maintain steady gaze positions across all directions, with minimal drift from the fixation point. The plotted data showed consistency with the high accuracy and low variability biometrics observed, and corroborated the system's sensitivity in detecting subtle physiological deviations, such as microsaccades, which remained within normal ranges for the healthy cohort (*Figure 14*).

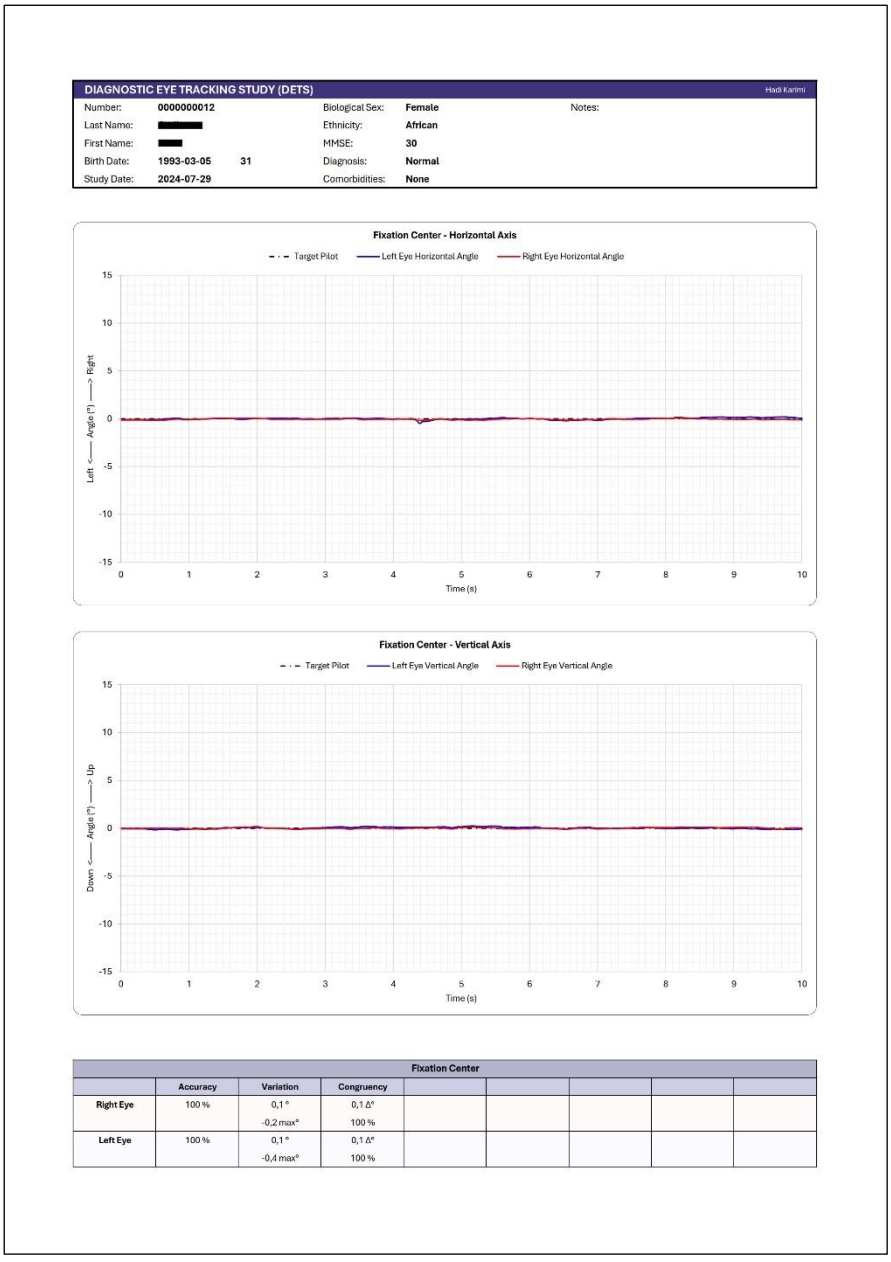


Figure 13: DETS Report; Fixation Center.

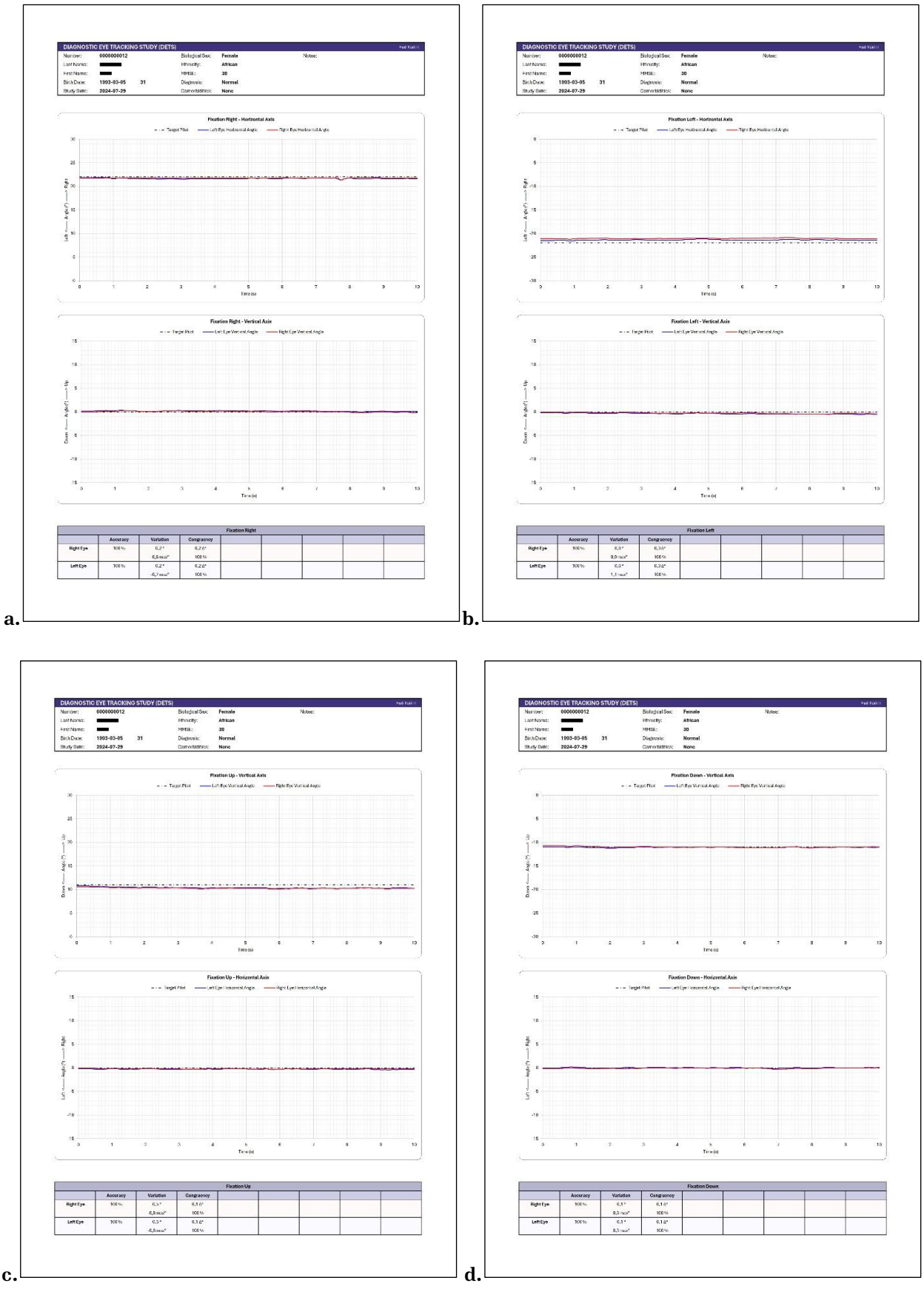


Figure 14: DETS report; a. Fixation Right, b. Fixation Left, c. Fixation Up, d. Fixation Down.

Table 4: Protocols Biometrics. SD, Standard Deviation; 99% CI, 99% Confidence Interval upper bound, lower bound, or both (range) accordingly

Biometrics	Accuracy			Variation			Congruency			Latency			Gain			Velocity			Phase Shift			Correct		
Protocols	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**	**Mean**	**SD**	**99% CI**
Fixation Central	98.91	1.75	97.32	0.13	0.06	0.19	0.25	0.13	0.37															
Fixation Right	98.57	1.96	96.78	0.17	0.13	0.29	0.35	0.11	0.45															
Fixation Left	98.71	2.78	96.18	0.15	0.08	0.23	0.41	0.18	0.57															
Fixation Up	98.59	2.26	96.53	0.19	0.16	0.33	0.37	0.15	0.50															
Fixation Down	98.90	2.52	96.60	0.14	0.08	0.21	0.43	0.13	0.55															
Saccades Horizontal	91.74	3.36	88.68	0.96	0.26	1.20	0.38	0.09	0.46	253.72	45.50	295.16	0.98	0.02	0.97 1.00	186.26	41.06	148.87						
Saccades Vertical	91.85	6.95	85.52	0.63	0.23	0.84	0.39	0.11	0.49	234.91	53.83	283.93	0.97	0.07	0.91 1.03	96.48	18.63	79.51						
Pursuit Horizontal	98.83	2.10	96.91	0.30	0.13	0.42	0.39	0.10	0.48				0.95	0.17	0.8 1.11				0.04	0.47	-0.38 0.47			
Pursuit Vertical	98.75	1.39	97.48	0.27	0.13	0.38	0.37	0.09	0.45				0.95	0.18	0.78 1.11				-0.39	1.16	-1.44 0.67			
OKN Right				1.06	0.62	1.62							0.95	0.21	0.76 1.14									
OKN Left				0.94	0.64	1.53							0.88	0.25	0.65 1.11									
OKN Up				1.26	0.54	1.76							0.81	0.17	0.65 0.96									
OKN Down				1.02	0.35	1.34							0.74	0.14	0.61 0.87									
ProSaccades Horizontal										523.66	129.95	642.01										3.94	1.95	2.16
ProSaccades Vertical										564.91	144.14	696.19										4.34	1.29	3.17
AntiSaccades Horizontal										634.72	147.59	769.14										5.25	0.57	4.73
AntiSaccades Vertical										624.47	131.57	744.30										4.34	1.10	3.35

Saccades

Horizontal saccades demonstrated a mean accuracy of 91.74% (SD = 3.36), with a 99% CI lower bound of 88.68%. The lower accuracy compared to fixation protocols is consistent with the more challenging nature of saccadic movements, where maintaining precision is inherently difficult. The standard deviation suggests moderate interindividual variability in saccadic accuracy. The mean variation in horizontal saccades was 0.96 degrees (SD = 0.26), with an upper bound of 1.20 degrees in the 99% CI, indicating that variability beyond this threshold could be clinically relevant. Congruency in horizontal saccades had a mean of 0.38 (SD = 0.09) and a 99% CI upper bound of 0.46, demonstrating reliable consistency with fixation protocols.

Horizontal saccadic latency, a critical parameter in assessing saccadic performance, showed a mean of 253.72 milliseconds (ms) (SD = 45.50) and an upper bound of 295.16 ms in the 99% CI. Latencies exceeding this upper bound may indicate clinically significant delays in saccade initiation. The gain analysis for horizontal saccades revealed a mean of 0.98 (SD = 0.02) with a 99% CI range from 0.97 to 1.00. Deviations from this range could suggest abnormal saccadic control, potentially due to feedback mechanism dysfunctions. The mean velocity of horizontal saccades was 186.26 degrees/sec (SD = 41.06), with a 99% CI lower bound of 148.87 degrees/sec, indicating that velocities below this threshold may reflect sluggish saccadic responses.

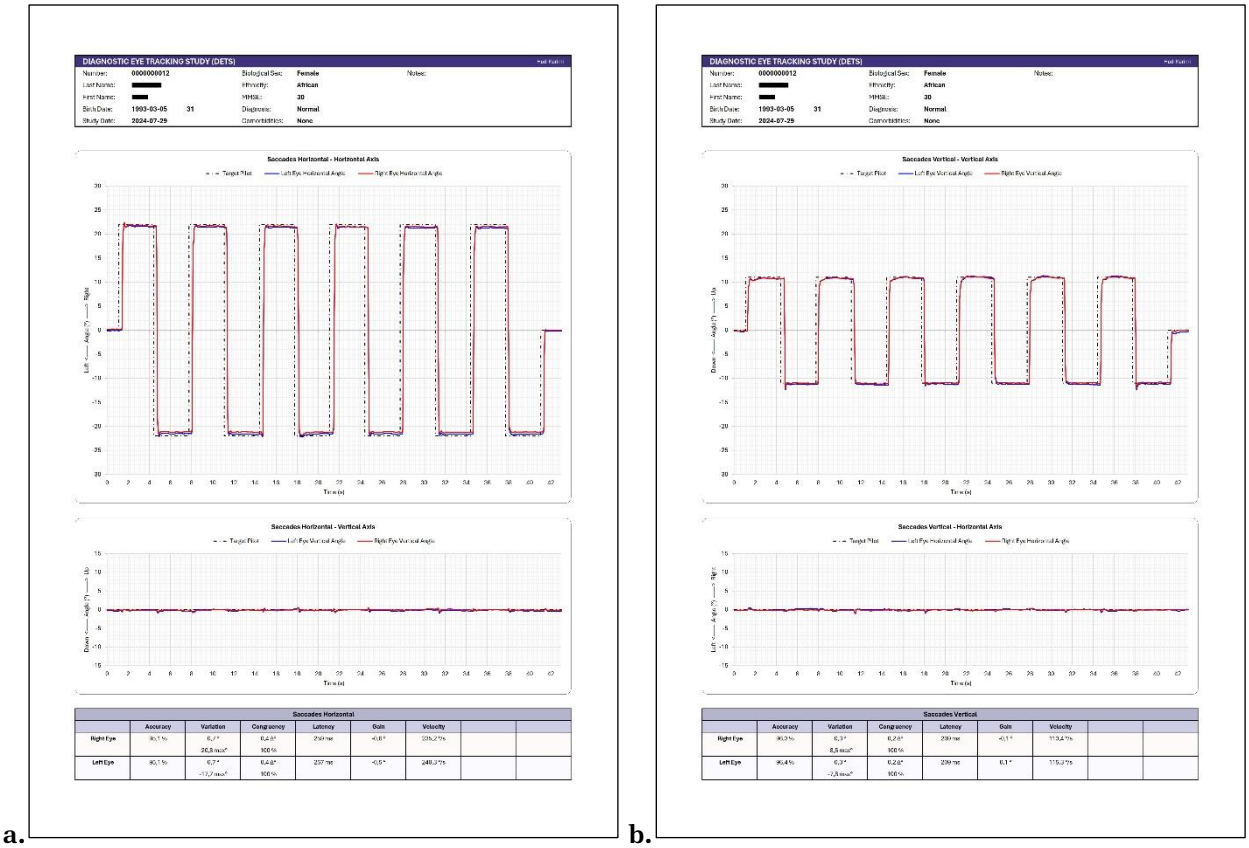


Figure 15: DETS report; a. Saccades Horizontal, b. Saccades Vertical.

Vertical saccades exhibited similar patterns, with a mean accuracy of 91.85% (SD = 6.95) and a 99% CI lower bound of 85.52%. Vertical saccadic latency averaged 234.91 ms (SD = 53.83), with an upper bound of 283.93 ms in the 99% CI. The mean velocity of vertical saccades was 96.48 degrees/sec (SD = 18.63), with a 99% CI lower bound of 79.51 degrees/sec. The reduced velocity in vertical saccades, compared to horizontal, aligns with the amplitude-dependent nature of saccadic velocity due to the shorter total displacement of the target stimulus in the vertical protocol. The mean gain for vertical saccades was 0.97 (SD = 0.07) with a 99% CI range from 0.91 to 1.03.

The saccade trajectory plots, capturing the rapid eye movements between fixed points, revealed sharp transitions typical of normal saccadic behavior. The time-series data corroborated the statistically observed latencies and velocities, confirming expected saccadic performance with minimal overshoot or undershoot, indicative of precise and accurate saccadic control (*Figure 15*).

Pursuit

Horizontal pursuit accuracy was high, with a mean of 98.83% (SD = 2.10) and a 99% CI lower bound of 96.91%, reflecting strong performance in tracking horizontal motion. The mean variation in pursuit was 0.30 degrees (SD = 0.13), with an upper bound of 0.42 degrees in the 99% CI, suggesting that greater variations may indicate difficulties in maintaining smooth pursuit. Congruency for horizontal pursuit was 0.39 (SD = 0.10) with an upper bound of 0.48 in the 99% CI. The gain for horizontal pursuit had a mean of 0.95 (SD = 0.17) and a 99% CI range from 0.81 to 1.11. Deviations from this range could be indicative of neurological deficits. The phase shift, a measure of the difference between eye movement timing and target motion, had a mean of 0.04 degrees (SD = 0.47) with a 99% CI range from -0.38 to 0.47 degrees, with significant deviations potentially reflecting abnormal pursuit dynamics such as catch-up saccades.

Vertical pursuit accuracy was similarly high, with a mean of 98.75% (SD = 1.39) and a 99% CI lower bound of 97.48%. The mean variation was slightly lower at 0.27 degrees (SD = 0.13), with an upper bound of 0.38 degrees in the 99% CI, indicating consistent vertical pursuit control. Congruency biometrics were aligned with horizontal pursuit, showing a mean of 0.37 (SD = 0.09) and an upper bound of 0.45 in the 99% CI. The gain for vertical pursuit averaged 0.95 (SD = 0.18) with a 99% CI range from 0.78 to 1.11. Phase shift exhibited a broader range (mean = -0.39 degrees SD = 1.16) with a 99% CI range from -1.44 to 0.67 degrees, possibly due to the shorter displacement of the stimulus.

Pursuit protocol graphs illustrated smooth and continuous eye movements closely tracking the moving target. The steady and consistent eye movements observed in both horizontal and vertical pursuit graphs confirmed the high accuracy and congruency biometrics found in the quantitative analysis, with phase shifts and minor variations remaining within physiological norms (*Figure 16*).

Figure 16: DETS report; a. Pursuit Horizontal, b. Pursuit Vertical.

Optokinetic Nystagmus (OKN)

OKN responses, measured in four directions (right, left, up, down), revealed gain values for horizontal OKN of 1.06 (SD = 0.62) for right and 0.94 (SD = 0.64) for left. The 99% CI ranges for right OKN were 0.76 to 1.14, and for left OKN were 0.65 to 1.11, indicating a broader normal range relative to saccades.

Vertical OKN responses, both upwards and downwards, demonstrated lower gain values with means of 1.26 (SD = 0.54) for up and 1.02 (SD = 0.35) for down. The 99% CI ranges for upward and downward OKN were 0.65 to 0.96 and 0.61 to 0.87, respectively, suggesting that while vertical OKN responses are generally consistent, abnormalities may still be present if these thresholds are exceeded.

OKN protocol visualizations depicted the characteristic nystagmus patterns, with slow eye movements followed by rapid resetting saccades, indicative of a healthy vestibulo-ocular reflex. Time-series analysis confirmed that the gain values, despite higher variability, remained within expected physiological ranges, with minimal deviations from the norm (*Figure 17*).

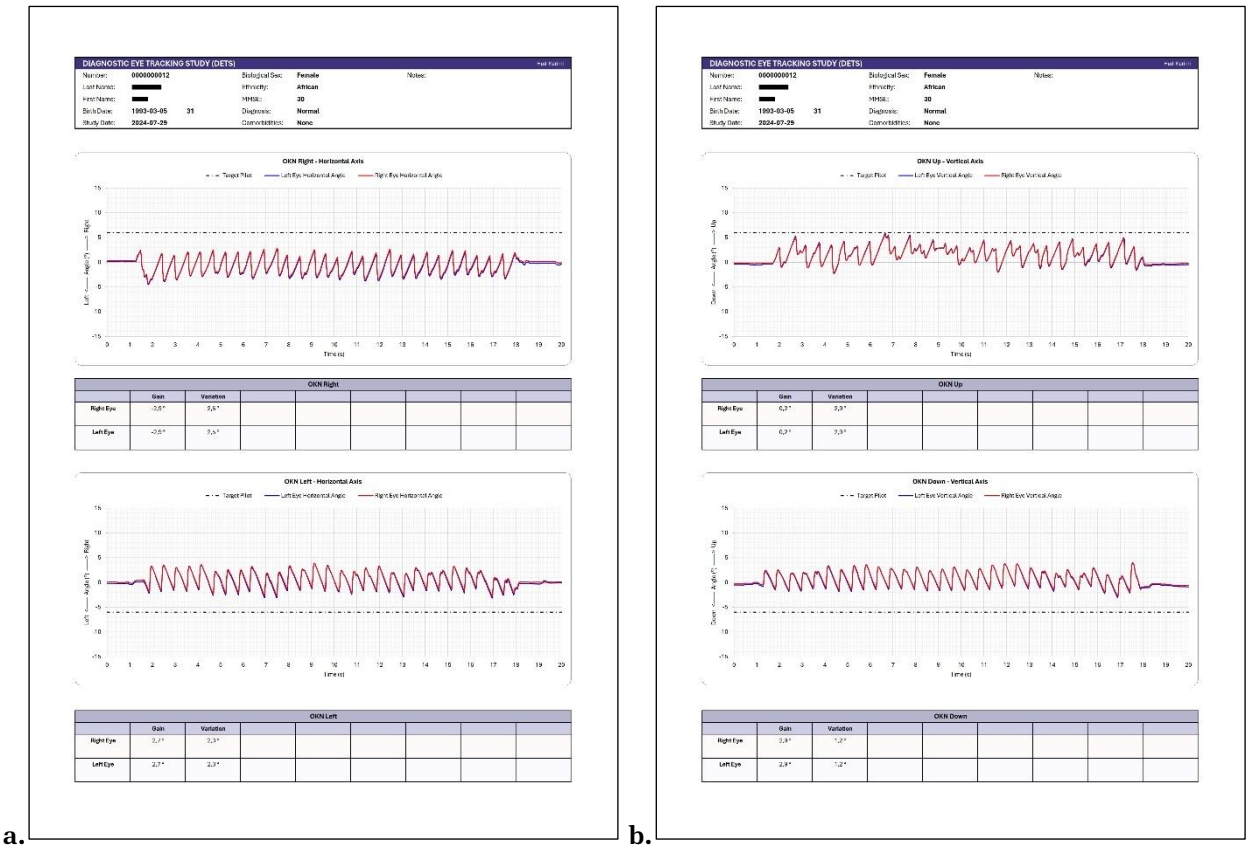


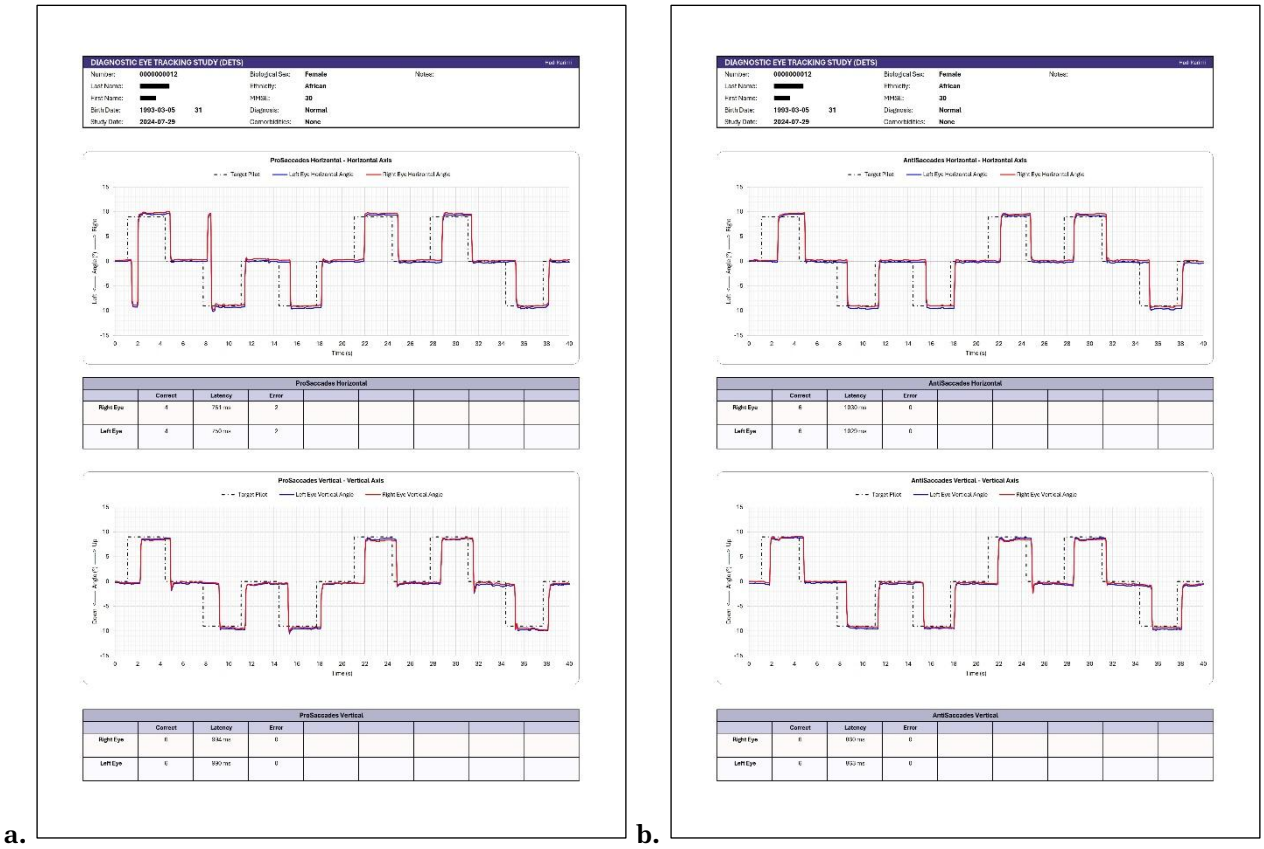
Figure 17: DETS report; a. OKN Horizontal (Right, Left), b. OKN Vertical (Up, Down).

Prosaccades and Antisaccades

The analysis of prosaccades and antisaccades provided insights into cognitive executive and inhibitory control. Horizontal prosaccades exhibited a mean latency of 523.66 ms (SD = 129.95), with an upper bound of 642.01 ms in the 99% CI, indicating that latencies beyond this threshold could reflect delayed saccadic initiation or processing. The correct rate for horizontal prosaccades was 3.94/6 (SD = 1.95), with a 99% CI lower bound of 2.16/6, suggesting suboptimal performance below this level. Vertical prosaccades showed a slightly longer latency (mean = 564.91 ms SD = 144.14) with an upper bound of 696.19 ms in the 99% CI. The correct rate was slightly higher at 4.34/6 (SD = 1.29), with a 99% CI lower bound of 3.17/6.

Horizontal antisaccades, requiring inhibition of a reflexive saccade and generation of a voluntary saccade in the opposite direction, exhibited a mean latency of 634.72 ms (SD = 147.59) with an upper bound of 769.14 ms in the 99% CI, reflecting the additional cognitive processing required for inhibitory control. The correct rate for horizontal antisaccades was 5.25/6 (SD = 0.57), with a 99% CI lower bound of 4.73/6. Vertical antisaccades showed a mean latency of 624.47 ms (SD = 131.57) and an upper bound of 744.30 ms in the 99% CI, with a correct rate of 4.34/6 (SD = 1.10) and a 99% CI lower bound of 3.35/6, suggesting that individuals with correct rates below this threshold may exhibit deficits in antisaccadic task performance.

Trajectory graphs for prosaccades and antisaccades highlighted differences between errors and errors with subsequent quick corrections. Combined with latency biometrics, these findings provide a comprehensive interpretation of cognitive and oculomotor control in subjects (*Figure 18*).



a. **b.**

Figure 18: DETS report; a. ProSaccade (Horizontal, Vertical), b. AntiSaccade (Horizontal, Vertical).

Patient Cases

During the optimization of protocols' stimuli and graphical illustrations of the final report, prior to initiating a healthy cohort study, we conducted Diagnostic Eye-Tracking Study (DETS) on a cohort of patients with established neurological conditions. This study integrated clinical findings and neuroimaging with detailed DETS results. While the in-depth case analysis yielded significant insights into the clinical utility of DETS and informed the optimization of protocols and report design, statistical analysis of the biometric data was not performed due to confounding factors, including the small sample size and the absence of appropriately matched controls. Nevertheless, qualitative analysis was undertaken, highlighting the potential of DETS to detect subtle eye-tracking abnormalities that were not clinically apparent. Below, we provide illustrative examples.

Friedreich's Ataxia with Macrosaccadic Oscillations

A 28-year-old male with a confirmed diagnosis of Friedreich's ataxia underwent Diagnostic Eye-Tracking Study (DETS). The study revealed macrosaccadic oscillations, which were initially misinterpreted clinically as opsoclonus. The distinguishing feature of macrosaccadic oscillations in this case was the presence of inter-saccadic intervals ranging from 100 to 500 milliseconds, evident as a flattening at the termination of saccades. This finding emphasized the value of DETS in refining clinical diagnoses, especially when ocular motor abnormalities are subtle or atypical (*Figure 19*).

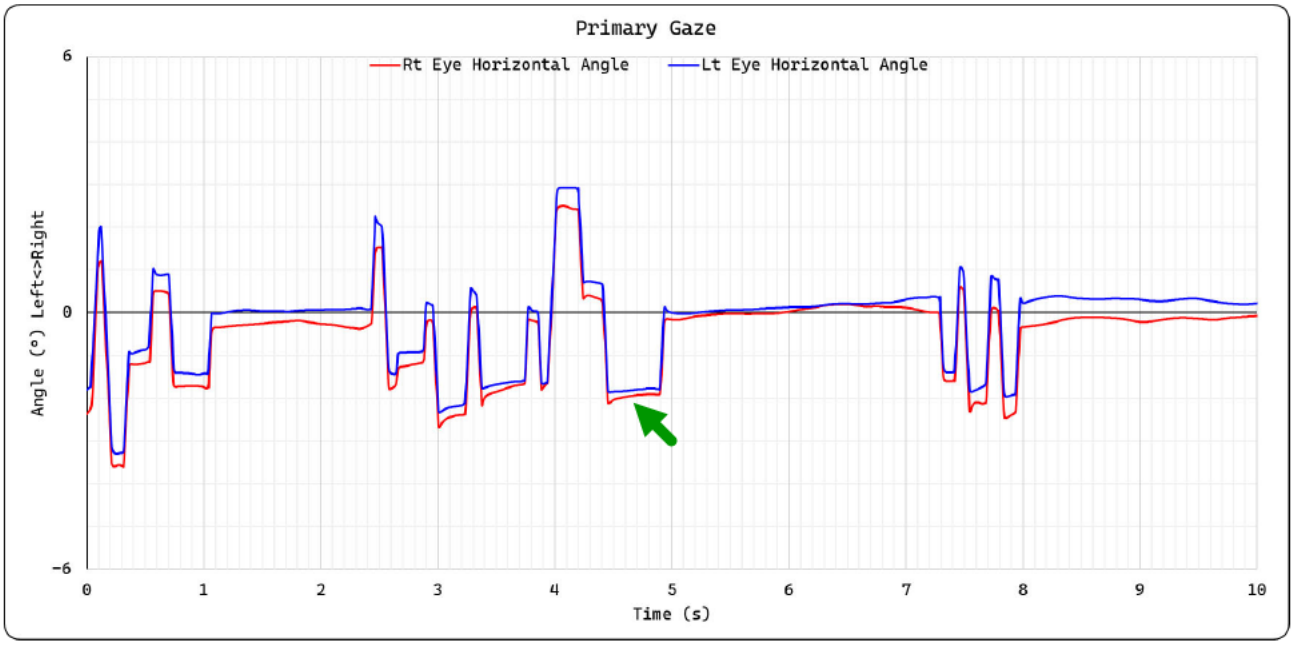


Figure 19: Macroscopic oscillations identified via DETS in a 28-year-old male with Friedreich's ataxia. The inter-saccadic intervals (100-500 ms) are noted as a flattening at the ends of saccades (green arrow), a key feature differentiating macrosaccadic oscillations from opsoclonus.

Myasthenia Gravis with Pseudo-INO

A 33-year-old female presented with bilateral adduction deficits, initially diagnosed as bilateral internuclear ophthalmoplegia (INO). However, DETS revealed intra-saccadic fatigue, suggesting a diagnosis of Pseudo-INO secondary to Myasthenia Gravis. This differentiation highlights the diagnostic precision afforded by DETS (*Figure 20*).

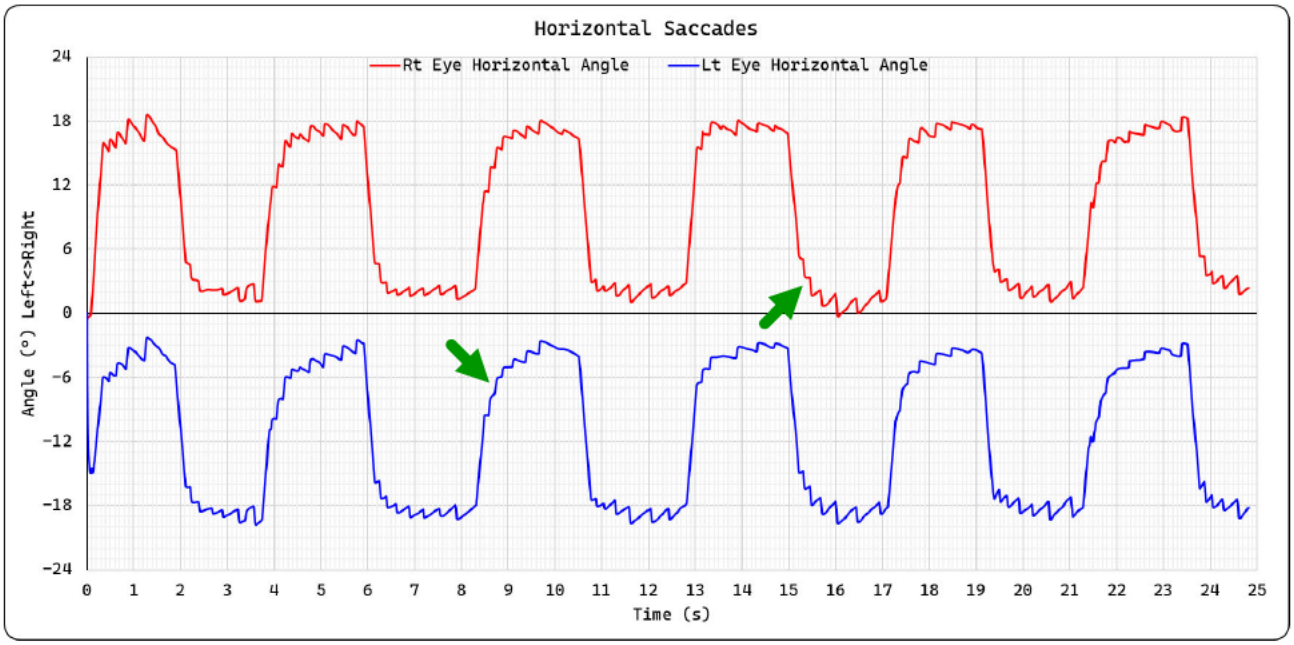


Figure 20: DETS findings in a 33-year-old female with Myasthenia Gravis. The image shows intra-saccadic fatigue (green arrows), which is commonly observed in Pseudo-INO compared to true bilateral INO.

Progressive Supranuclear Palsy with Vertical Gaze Instability

A 68-year-old male diagnosed with Progressive Supranuclear Palsy (PSP) presented with postural instability and classic Parkinson's syndrome. DETS provided additional insights, revealing significant instability in vertical gaze, particularly in slow upward pursuit and saccadic interruptions during downgaze. These findings align with the characteristic ocular motor abnormalities seen in PSP and underscore the utility of DETS in detecting these subtle changes (*Figure 21*).

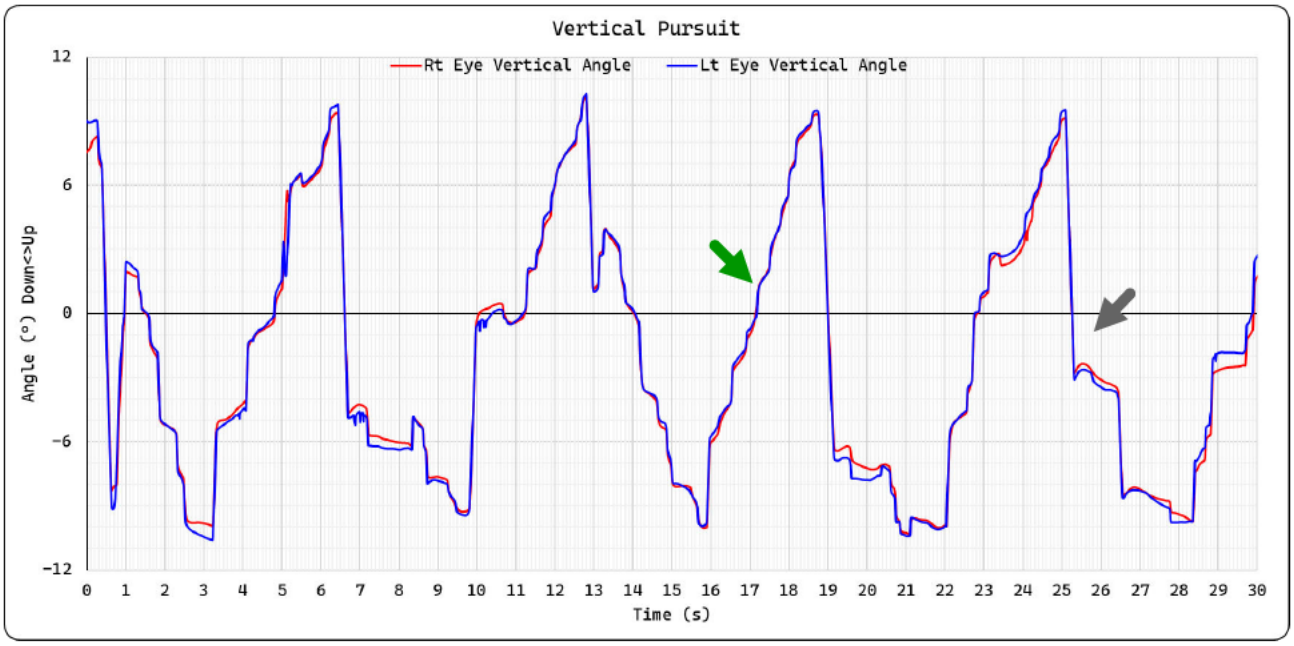


Figure 21: DETS analysis of a 68-year-old male with PSP. Notable findings include vertical gaze instability, specifically slow upward pursuit (green arrow) and saccadic interruptions during downgaze (gray arrow), consistent with PSP.

Hemorrhagic Stroke with Cerebellar Involvement

A 43-year-old female of African descent with a history of uncontrolled hypertension presented with acute onset of dizziness and dysarthria. Neurological examination revealed bilateral cerebellar signs, more pronounced on the right, including dysmetria and gait impairement. Given the patient's history and the acute nature of her symptoms, an MRI was promptly conducted (*Figure 22).*

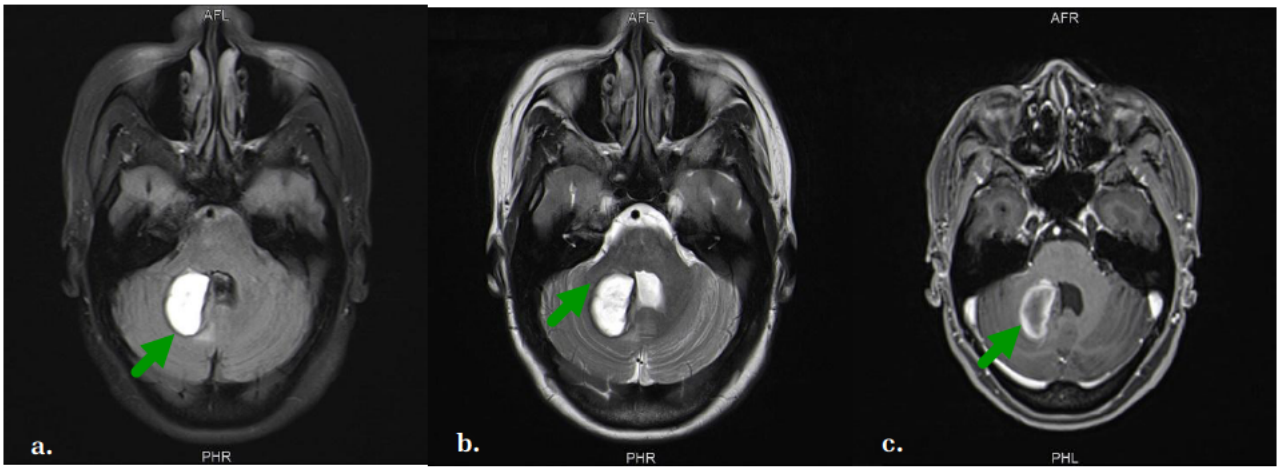
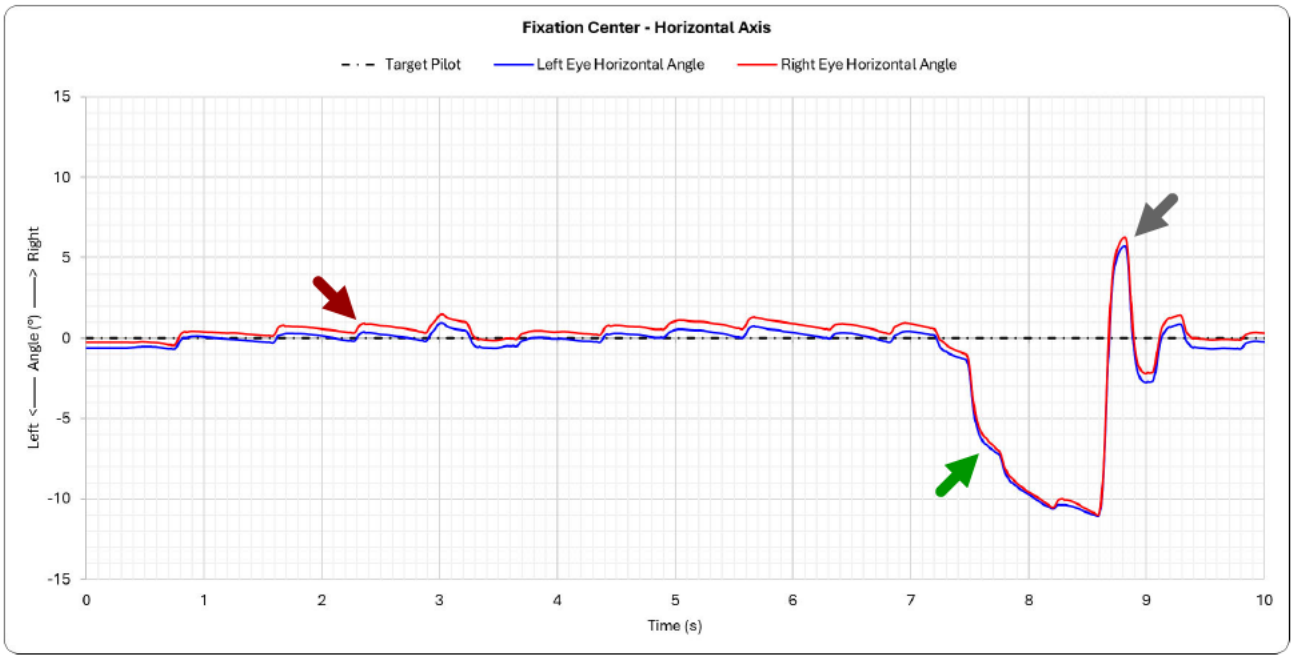


Figure 22: MRI imaging of a hemorrhagic lesion in the posterior fossa, predominantly involving the right cerebellum. a, Axial FLAIR MRI showing hyperintensity consistent with hemorrhagic stroke. b, Axial T2-weighted MRI demonstrating the involvement of the right middle cerebellar peduncle. c, Axial T1-weighted post-contrast MRI highlighting the hemorrhagic component.

MRI imaging revealed a hemorrhagic stroke localized to the right paracentral cerebellum, specifically impacting the right middle cerebellar peduncle. The lesion was significant for its effect on the cerebellar circuitry responsible for coordinating horizontal gaze.

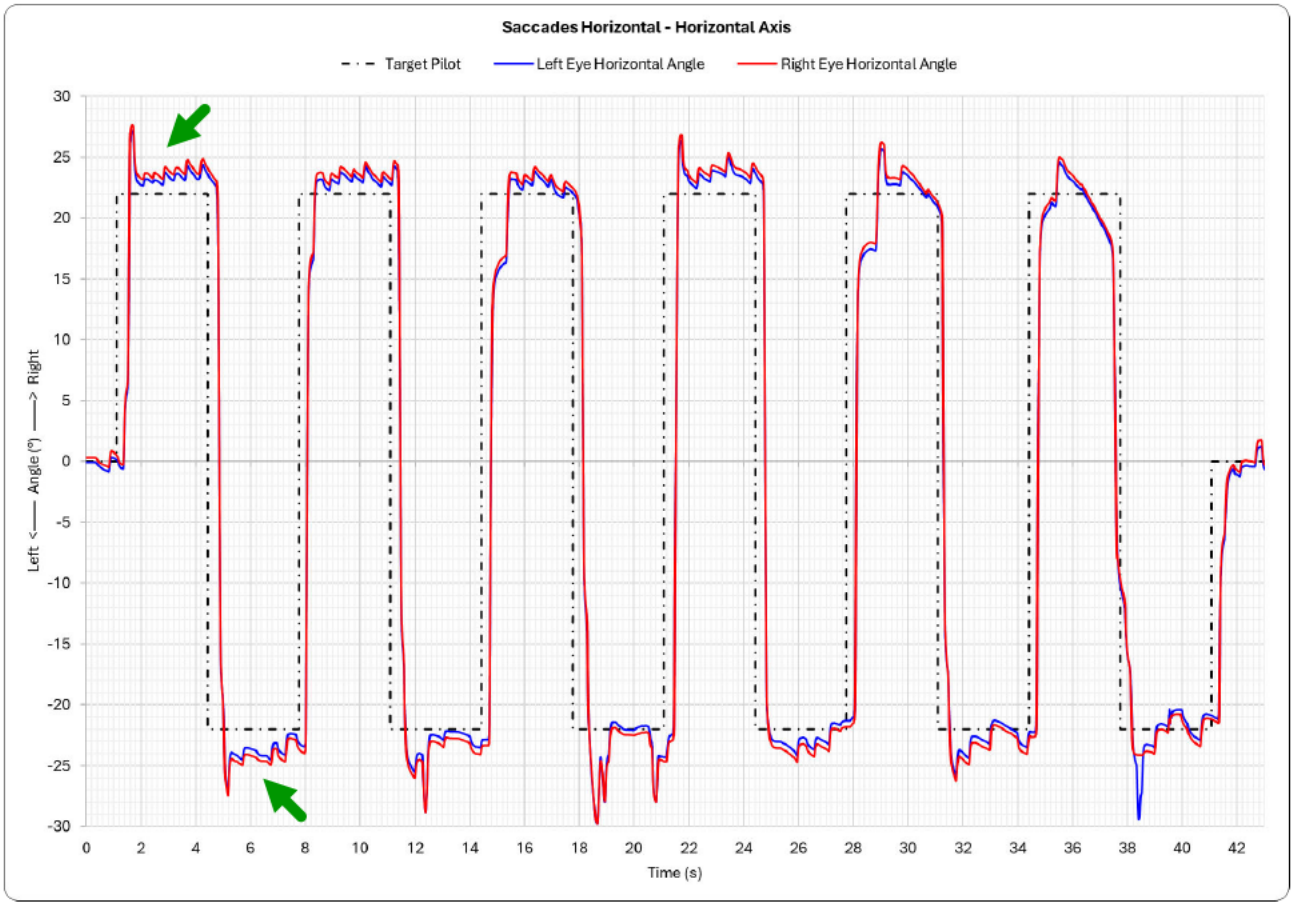
In the context of horizontal gaze network impairment, the DETS results were consistent with the neuroimaging findings. The Fixation Central report highlighted the presence of a right-beating nystagmus with a leftward drift, followed by a hypermetric corrective saccade with overshoot (*Figure 23*). This pattern is indicative of disrupted horizontal gaze stabilization, directly correlating with the right-sided cerebellar lesion identified on MRI.



Fixation Center								
	Accuracy	Variation	Congruency					
Right Eye	76,6 %	1,3 °	0,6 Δ°					
		-10,2 max°	99,4 %					
Left Eye	77,1 %	1,3 °	0,6 Δ°					
		-9,8 max°	99,4 %					

Figure 23: DETS Fixation Central report showing a right-beating nystagmus with a leftward drift and hypermetric correction (red arrow), consistent with the cerebellar lesion identified on MRI. The green arrow highlights the slow drift, while the gray arrow shows saccadic oscillations.

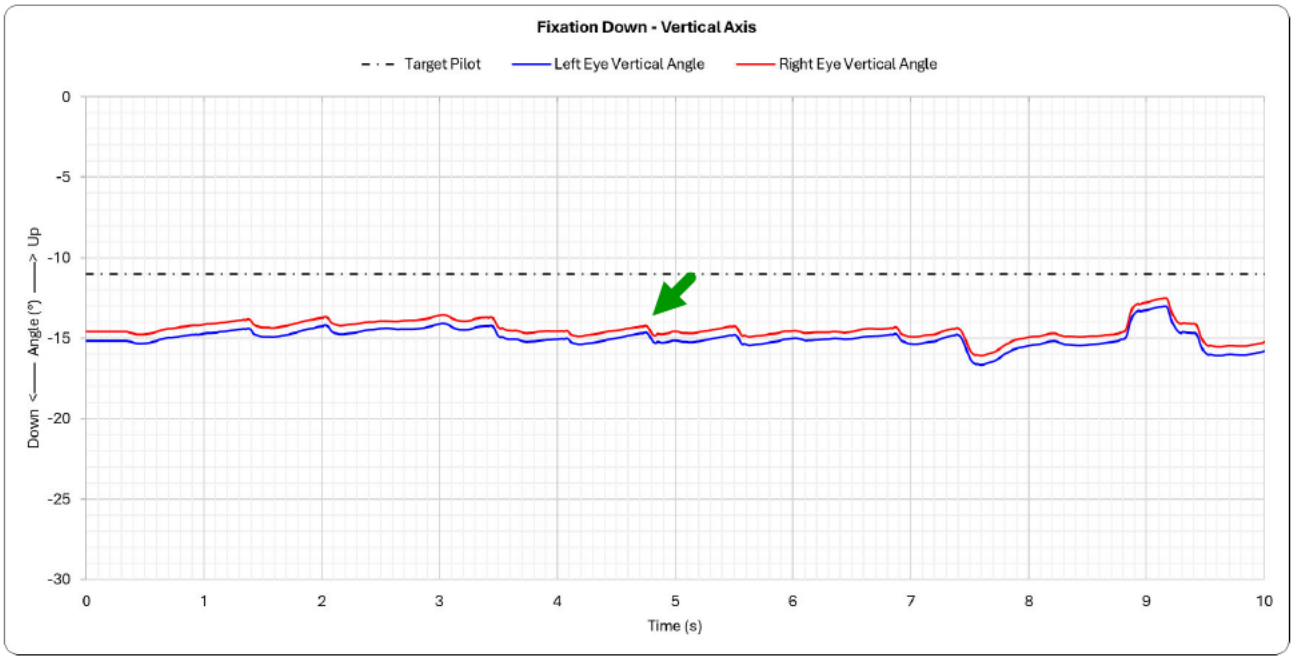
Further examination of the Horizontal Saccades report confirmed the persistence of right-beating nystagmus at fixation points, with abnormalities in gain values (*Figure 24*). These findings suggest a disruption in cerebellar control mechanisms that are involved in modulating saccadic gain and ensuring precise eye movements. Despite the abnormal gain, both saccade latency and velocity remained within normal limits, indicating that the superior colliculus, responsible for initiating saccades and controlling their velocity, was not affected by the lesion, consistent with the imaging studies.



Saccades Horizontal								
	Accuracy	Variation	Congruency	Latency	Gain	Velocity		
Right Eye	58,3 %	1,8 °	0,6 Δ°	266 ms	0,95	163,7 °/s		
		-21,9 max°	99,4 %					
Left Eye	71 %	1,6 °	0,6 Δ°	262 ms	0,95	159 °/s		
		-21,6 max°	99,4 %					

Figure 24: Horizontal Saccades report indicating persistent right-beating nystagmus and ocular dysmetria at fixation points. Abnormal gain values are evident, indicating impaired cerebellar modulation of saccadic accuracy, although latency and velocity remained within normal ranges.

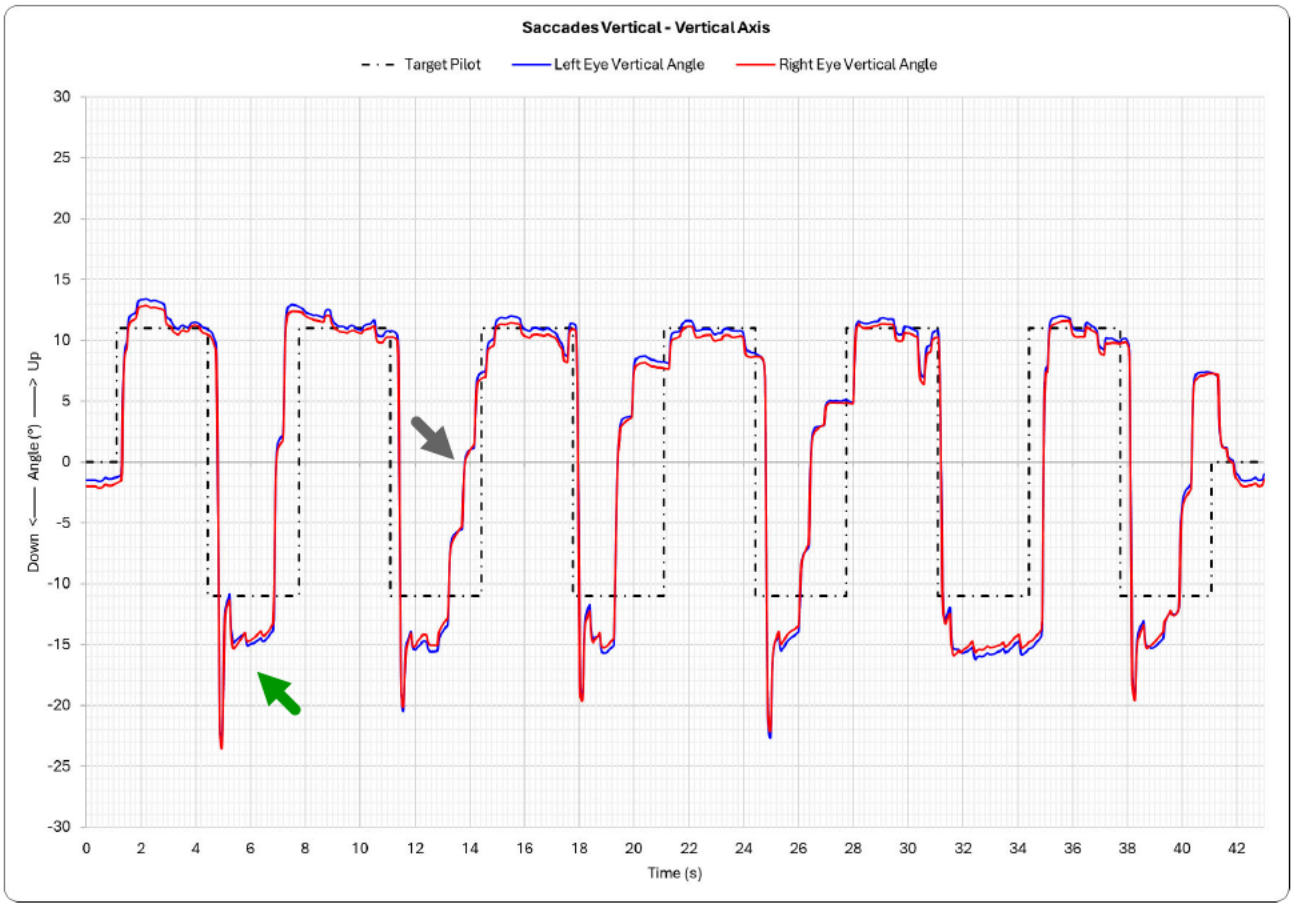
An intriguing finding emerged in the Fixation Down report, where DETS detected a subtle down-beating nystagmus. This nystagmus was not clinically apparent during bedside examination, highlighting the enhanced sensitivity of DETS in identifying subtle ocular motor abnormalities. The detection of such nuances may indicate more widespread dysfunction than initially suspected (*Figure 25*).



Fixation Down								
	Accuracy	Variation	Congruency					
Right Eye	95,9 %	0,3 °	0,6 Δ°					
		1,4 max°	100 %					
Left Eye	95,4 %	0,4 °	0,6 Δ°					
		1,7 max°	100 %					

Figure 25: Fixation Down report identifying a downgaze fixation instability off the pilot target position (black dotted line) and a subtle down-beating nystagmus (green arrow) during downgaze.

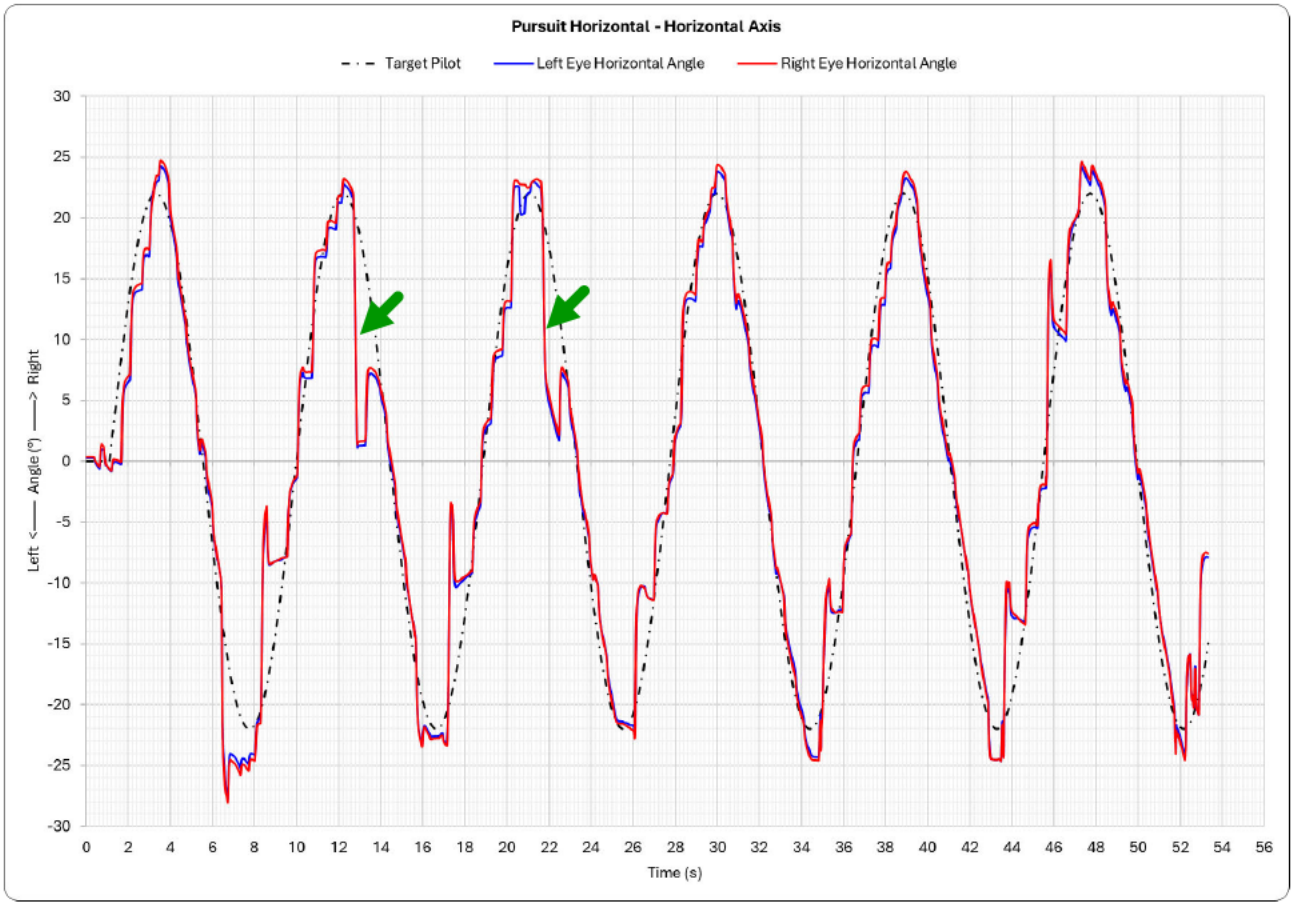
The Vertical Saccades report further expanded on these findings, revealing significant downward overshoot and nystagmus when the patient attempted to fixate on a downward target. These hypermetric saccades were accompanied by intra-saccadic inconsistencies, particularly during upward movements. The prolonged latency observed in downward saccades, despite maintaining normal velocity, suggests a deficit in the initiation phase of saccadic movements. This pattern in combination with normal horizontal saccades latency indicates a disruption in the infratentorial vertical saccade pathways, particularly those involving the rostral interstitial nucleus of the medial longitudinal fasciculus (riMLF) and its connections (*Figure 26*).



Saccades Vertical								
	Accuracy	Variation	Congruency	Latency	Gain	Velocity		
Right Eye	53,2 %	2 °	0,6 Δ°	363 ms	2,23	104,1 °/s		
		15,4 max°	99,4 %					
Left Eye	51,8 %	2,1 °	0,6 Δ°	378 ms	2,07	110,9 °/s		
		16,1 max°	99,4 %					

Figure 26: Vertical Saccades report showing a significant downward overshoot (green arrow) with nystagmus, and intra-saccadic inconsistencies during upward movements (gray arrow).

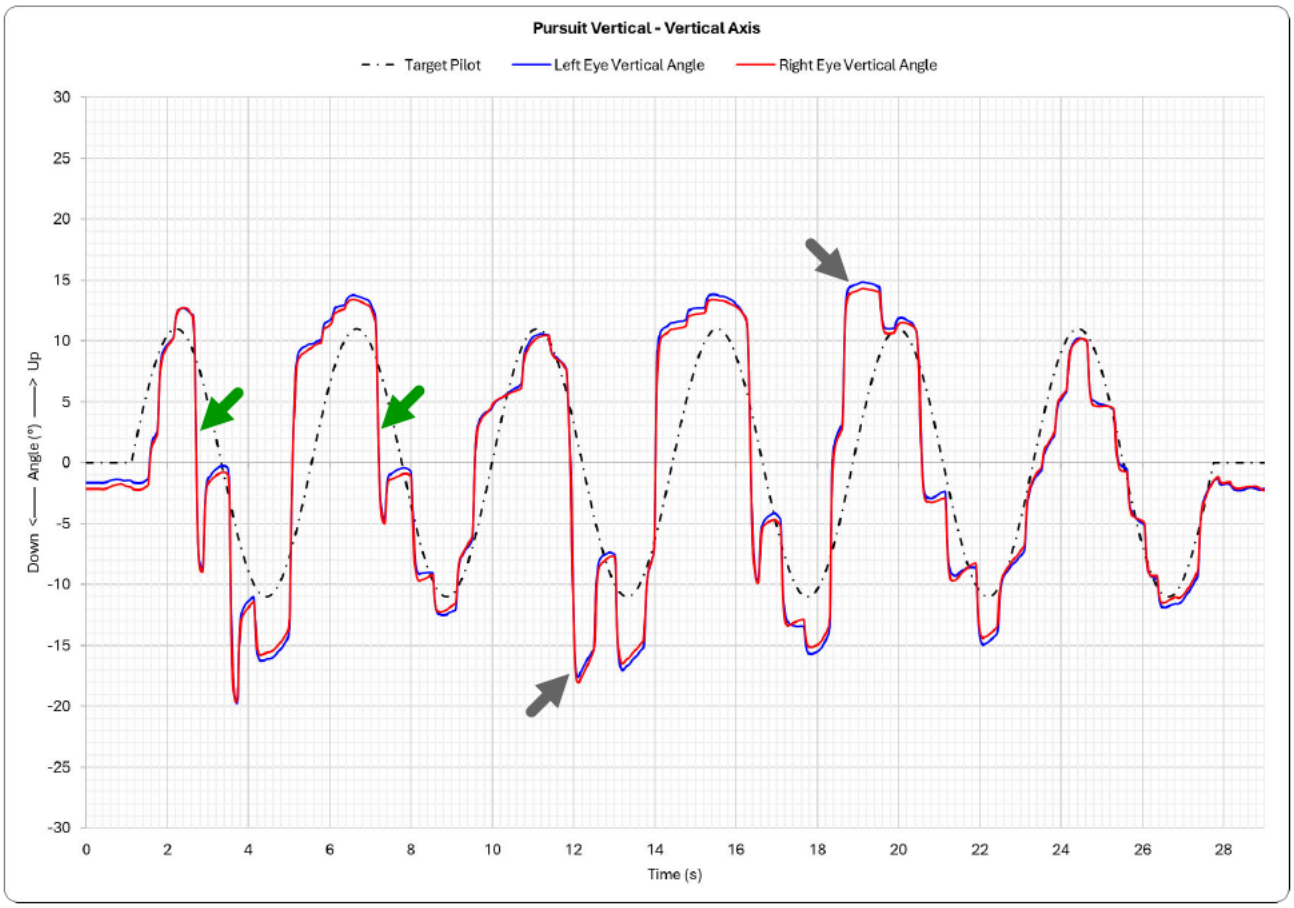
The Horizontal Pursuit report showed the presence of catch-up saccades, which are consistent with impaired smooth pursuit mechanisms, a known consequence of cerebellar damage. This finding aligns with the cerebellar lesion observed on MRI and provides a functional correlate to the structural damage (*Figure 27*).



Pursuit Horizontal								
	Accuracy	Variation	Congruency	Phase Shift	Gain			
Right Eye	73,5 %	1,1 °	0,5 Δ°	0,3 °	1,29			
		-7,2 max°	99,1 %					
Left Eye	82 %	1 °	0,5 Δ°	-0,1 °	1,29			
		-6,6 max°	99,1 %					

Figure 27: Horizontal Pursuit report showing catch-up saccades (green arrows), indicating impaired smooth pursuit mechanisms associated with the cerebellar lesion.

The Vertical Pursuit report revealed more pronounced catch-up saccades, which were accompanied by a substantial increase in gain. This finding suggests a broader impairment involving both tonic and inhibitory neural pathways involved in vertical gaze control. The increased gain indicates that the damage extends beyond the cerebellum, likely involving other brainstem structures responsible for vertical gaze stabilization (*Figure 28*).



Pursuit Vertical								
	Accuracy	Variation	Congruency	Phase Shift	Gain			
Right Eye	76,8 %	1 °	0,7 Δ°	0,9 °	8,75			
		-7,1 max°	95,7 %					
Left Eye	71,7 %	1,1 °	0,7 Δ°	1,2 °	8,82			
		-8,6 max°	95,7 %					

Figure 28: Vertical Pursuit report showing pronounced catch-up saccades (green arrows) and increased gain of ocular dysmetria (gray arrows).

Further analysis of the Optokinetic Nystagmus (OKN) in vertical directions (up/down) revealed severe impairment characterized by a significant increase in both gain and variation. This impairment aligns with the presence of pathology extending beyond the cerebellum and middle cerebellar peduncle, likely involving the brainstem.

Additionally, the Vertical AntiSaccade report revealed a significantly prolonged latency in the initiation of vertical saccades, despite the patient achieving an acceptable correct response rate. This pattern suggests that while the patient's higher cognitive functions, particularly those related to decision-making and inhibitory control (mediated by the frontal lobes) remained largely intact, there was a distinct disruption in the neural pathways responsible for the initiation of vertical eye movements.

Imaging corroborated these findings by revealing multiple smaller lesions at the level of the higher pons and lower midbrain (*Figure 29*). These regions are integral to the control of vertical saccades, particularly through pathways connecting the superior colliculus to vertical gaze nuclei, such as the riMLF. This correlation between DETS findings and neuroimaging highlights the importance of this integrated approach to achieve a comprehensive understanding of a patient's neurological deficits.

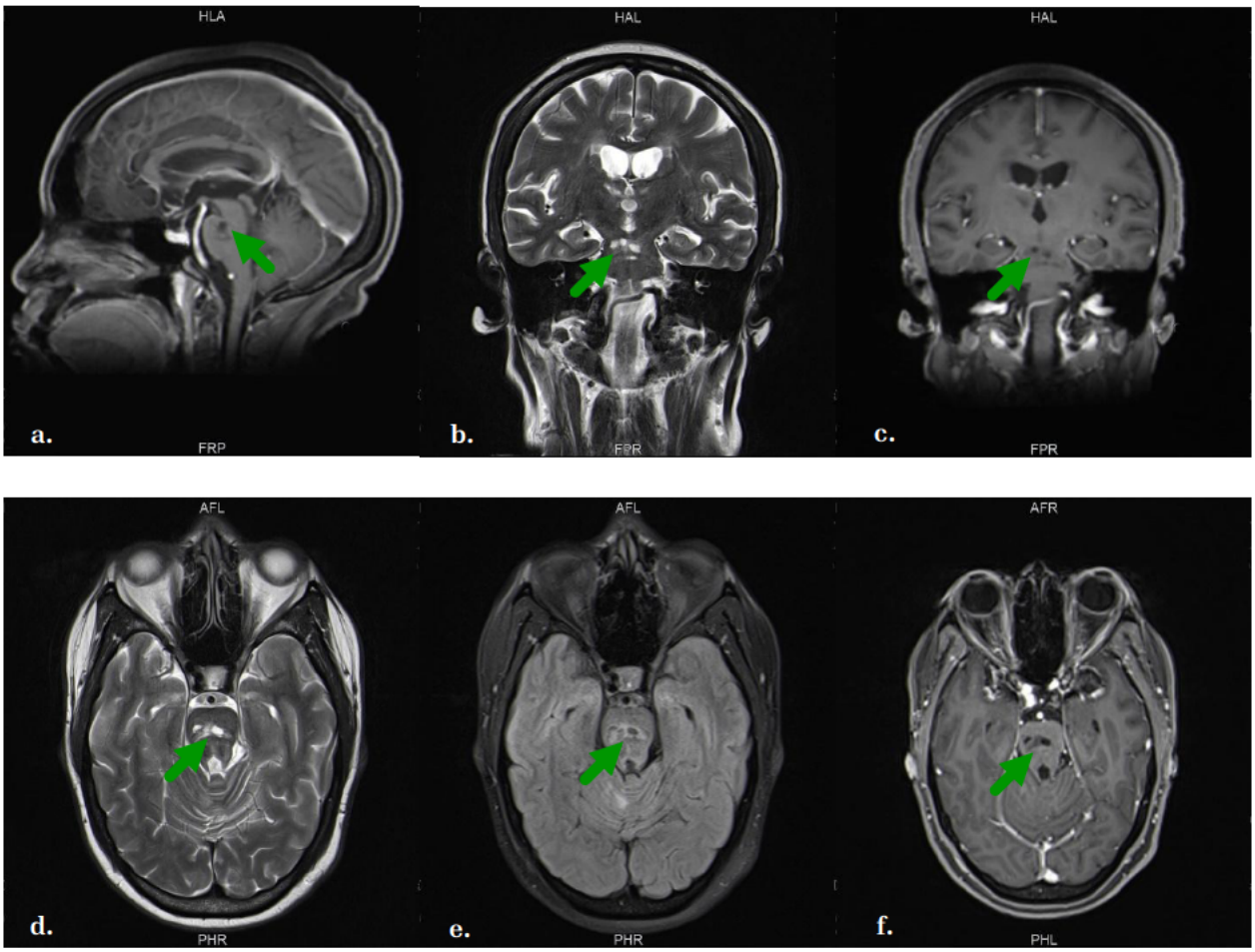


Figure 29: MRI of the lower midbrain/upper pontine tegmentum showing a lesion involving vertical gaze pathways (green arrows). Midsagittal T1-weighted post-contrast (a), Coronal T2-weighted (b), Coronal T1-weighted post-contrast (c), Axial T2-weighted (d), Axial FLAIR (e), and Axial T1-weighted post-contrast (f) MRI showing the lesion extent across the midbrain, illustrating the lesion's correlation with DETS findings and its impact on the vertical gaze control network.

Discussion

This study represents a pioneering effort to integrate the Diagnostic Eye-Tracking Study (DETS) as a diagnostic tool in routine clinical practice. Existing eye-tracking methodologies have been constrained by their narrow focus, lack of granularity, and cumbersome data analysis processes, limiting their applicability in routine clinical settings. These systems often focus on specific conditions with limited aspects of eye movement assessment, and necessitate manual, case-by-case processing, calculation, and analysis, which is impractical in routine clinical settings. In contrast, DETS was designed to meet the specific needs of clinical neurology, offering a comprehensive battery of protocols that capture a wide spectrum of eye movements, along with automated data processing, analysis, and biometric calculations. This approach not only enhances diagnostic accuracy but also streamlines the integration of eye-tracking into clinical practice.

DETS was developed with a multidisciplinary approach, integrating advancements in clinical neurology, mathematics, physics, and software engineering. One of the key innovations of the DETS project is its ability to preserve the unique characteristics and integrity of subtle ocular motor aberrations through advanced signal processing algorithms such as dynamic window-size Savitzky-Golay filter. This ensures that subtle yet diagnostically significant abnormalities, such as minute intrusions that are often easily masked by artifacts, are accurately captured and analyzed. The ability to differentiate between genuine pathological signals and artifacts is particularly important in clinical settings, where reliable diagnosis is critical. However, existing methods, which often rely on static filtering techniques, tend to over-smooth data, potentially obscuring subtle but clinically significant eye movement abnormalities.

Furthermore, DETS prioritizes automation and scalability, ensuring that its implementation in clinical practice does not require extensive manual processing, which is a major limitation of many current systems. By automating data processing and analysis, DETS reduces the likelihood of human error and ensures consistent and reliable results across different clinical environments, thereby extending its benefits to a broad patient population and improving the overall quality of neurological care. Additionally, when compared with generic eye-tracking software, such as the system evaluated by Ehinger et al. (2019) (*Figure 30*), DETS showed superior performance.[166] While alternative platforms are more suited to research applications, the ability of DETS to produce clinically relevant results offering a broader range of protocols and automated analysis with minimal manual intervention set DETS apart from other systems.

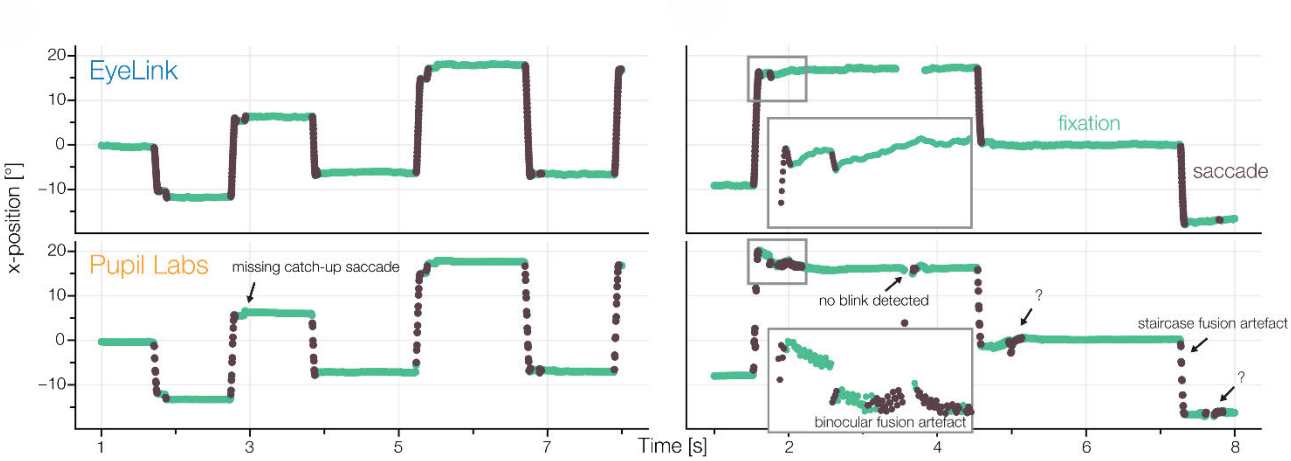


Figure 30: Comparison to results provided by generic software.[166]

The validation of the DETS software was conducted through a limited pilot study including both healthy participants and patients with neurological disorders. The DETS pilot study demonstrated a data loss rate of only 1.6%, which is notably low compared to other studies in clinical settings. For instance, Niehorster et al. (2018) reported that data loss in typical eye-tracking studies can range from 5% to 20%, particularly in less controlled environment.[167]

Additionally, studies have shown that eye-tracking data can be highly sensitive to noise and artifacts, particularly in older populations. The study by Opwonya et al. (2023) highlights the challenges of using eye-tracking in elderly patients, where data loss and artifacts are often exacerbated by difficulties in maintaining steady gaze and cognitive decline.[168] The study by Leharanger et al. (2024) showed that careful post-processing can significantly reduce artifact rates, similar to what was achieved in the DETS study, where artifact rates were reduced from 11.7% to 4.2%.[169] This demonstrates the effectiveness of the denoising algorithms used in DETS, which maintained data integrity across different populations and in a clinical settings where patient compliance were variable.

The enhancement in signal-to-noise ratio (SNR) is particularly important as a high SNR ensures that the true signal is not overshadowed by noise and retains high quality for analysis. The study also evaluated the extent to which the signal was distorted during processing. The distorted signal rate was found to be extremely low (0.01%), which is a noteworthy achievement that indicates the denoising process preserved the integrity of the original signal, and therefore the diagnostic information derived from the data is both accurate and reliable. The study also employed statistical validation techniques, including the Kolmogorov-Smirnov test, which confirmed that the data processing steps did not introduce any significant alterations in the data distribution.

DETS incorporates a comprehensive protocol battery composed of 17 distinct protocols to assess different aspects of eye movement, ensuring that the stimuli and tasks are clinically relevant and capable of revealing subtle abnormalities in eye movements. The DETS application generates a comprehensive report that includes both quantitative biometrics and qualitative graphs by plotting

gaze angle trajectory over time, providing fine visual representations of eye movement patterns and abnormalities to facilitate the identification of unique eye movement signatures associated with various neurological disorders.

The fixation protocols in the DETS software were designed to assess gaze stabilization by requiring participants to fixate on a static target positioned at various angles in the visual field. The goal was to evaluate the subject's ability to maintain a steady gaze across different directions. Across healthy participants, the fixation protocols demonstrated consistently high accuracy across all tested directions. For instance, the central fixation accuracy had a mean of 98.91% with a 99% confidence interval (CI) lower bound of 97.32%. The accuracy in other directions—right, left, up, and down—was similarly high, with 99% CI lower bounds ranging from 96.18% to 96.78%. These findings align closely with normative values reported in the literature, such as those by Ionescu et al. (2023) and Andersson et al. (2017), indicating that fixation accuracy in healthy individuals typically hovers around 98.90%, particularly in controlled settings where external distractions are minimized.[19,34]

Saccadic protocols were integral to the DETS assessment, focusing on task-specific eye movements that require rapid shifts in gaze between two peripheral targets, testing both the speed and accuracy of these movements. In healthy individuals, the mean accuracy for horizontal saccades was 91.74%, with a 99% CI lower bound of 88.68%, while vertical saccades had a slightly lower mean accuracy of 91.85%, with a 99% CI lower bound of 85.52%. Saccadic latency was also within expected norms, with mean latencies of 253.72 ms for horizontal saccades (99% CI upper bound: 295.16 ms) and 234.91 ms for vertical saccades (99% CI upper bound: 283.93 ms). Studies by Leigh and Zee (2015) and Anderson and MacAskill (2013) report similar latency ranges for saccadic movements, typically between 150-250 milliseconds in healthy adults.[1,34] Additionally, the saccadic gain in these participants was close to the ideal value of 1.0, with a 99% CI between 0.97 and 1.00. This is in line with the expected performance in healthy populations, as documented by Imaoka et al. (2020), who found that saccadic gain typically remains close to 1.0 under normal conditions.[170]

However, the saccadic velocity measurements presented some discrepancies. The mean velocity for horizontal saccades was 186.26 degrees per second with a 99% CI lower bound of 148.87 degrees per second. For vertical saccades, the mean velocity was notably lower at 96.48 degrees per second, with a 99% CI lower bound of 79.51 degrees per second. These values are lower than the broader range typically reported in the older literature (1974-1988), where saccadic velocities are usually between 300 and 700 degrees per second.[171 173] The discrepancy can be attributed to the amplitude-dependent nature of saccadic velocity, where larger saccades generally exhibit higher velocities, a relationship well-documented in the "main sequence" of saccades by Guadron et al. (2022), Gibaldi et al. (2021), and Anderson and MacAskill (2013).[34,109,174] The saccades tested in the DETS protocols were of moderate amplitude, with a total displacement of 44 degrees for horizontal saccades and 22 degrees for vertical saccades. This moderate amplitude likely accounts for the lower velocity measurements. In studies where larger saccadic amplitudes are used, higher velocities are typically observed, supporting the notion that the velocity of a saccade is not fixed

but scales with the amplitude of the eye movement. This relationship should be considered when comparing velocity measurements across different studies.

The pursuit protocols in the DETS evaluated the ability of participants to smoothly follow a moving target with their eyes, with pursuit gain and phase shift serving as critical performance biometrics. Among the healthy participants, the horizontal pursuit exhibited a mean accuracy of 98.83%, with a 99% CI lower bound of 96.91%, while the vertical pursuit demonstrated a mean accuracy of 98.75%, with a 99% CI lower bound of 97.48%. These high levels of accuracy are consistent with what has been documented in similar studies. In terms of pursuit gain, DETS recorded values that closely approximated 1.0, a benchmark indicating ideal performance. Specifically, the 99% CI for horizontal pursuit gain ranged from 0.81 to 1.11, while for vertical pursuit gain, the CI ranged from 0.78 to 1.11. The high accuracy and narrow confidence intervals in pursuit Protocols observed in this study align with the findings of recent studies.[175,176] Moreover, Lima et al. (2023) highlighted the importance of minimizing noise in achieving accurate pursuit gain measurements with consistency of close to 1.0 in healthy populations when measured under optimal conditions.[175] These comparisons not only reinforce the validity of the DETS findings but also position the system as a reliable tool for assessing smooth pursuit eye movements.

The Optokinetic Nystagmus (OKN) protocols implemented in the DETS application were designed to assess reflexive eye movements in response to a moving visual field. The results from healthy participants in the study provided robust data that align well with established physiological norms. For horizontal OKN, tracking rightward motion and leftward motion, the 99% CI ranged from 0.76 to 1.14, and for vertical OKN, tracking upward motion and downward motion, the 99% CI ranged from 0.61 to 0.96. These results showed slightly wider range for vertical OKN, yet consistent with previous findings by Leigh and Zee (2015) and more recent study by Lemos et al. (2021), which also reported similar directional discrepancies in OKN gain, particularly noting the tendency for vertical OKN gains to be slightly higher.[1,177] Additionally, Norouzifard et al. (2020) reported that OKN gain can exhibit slight variations due to factors such as age or fatigue. The study also noted that narrow CIs, as seen in DETS, are indicative of the reliability and precision of the OKN measurements, underscoring the robustness of the DETS application.[178]

The ProSaccades and AntiSaccades protocols in DETS were designed to evaluate the participants' abilities to execute directed saccades toward a target (ProSaccades) and to inhibit a reflexive saccade in favor of a voluntary saccade in the opposite direction (AntiSaccades). These tasks are used in assessing cognitive functions such as attention, executive function, and response inhibition. In the ProSaccades protocols, the mean latency for the horizontal protocol was observed to be slightly lower than the vertical protocol, with a 99% CI upper bound of 642.01 ms for horizontal and 696.19 ms for vertical. This discrepancy, although unexpected, aligns with findings from other studies that have noted the impact of vertical versus horizontal saccades on latency times.[179,180] The observed lower correct response rates for horizontal ProSaccades, with a 99% CI lower bound of 2.16 from 6, compared to 3.17 from 6 for vertical ProSaccades, could be attributed to the sequence in which the tasks were administered and the lack of practice sessions, potentially influencing the participants' performance. Hutton (2008) and Ettinger et al. (2003)

reported this variability in response times and accuracy based on task difficulty and participant readiness.[181,182]

For the AntiSaccades protocols, the horizontal protocol showed a 99% CI upper bound latency of 769.14 ms, while the vertical protocol had a slightly shorter latency with a 99% CI upper bound of 744.30 ms. These results are consistent with normative values reported by Munoz and Everling (2004), who found that AntiSaccades typically exhibit longer latencies due to the cognitive demands of inhibiting a reflexive saccade and initiating a voluntary movement in the opposite direction.[183] The correct response rates in the DETS study, 5.25 from 6 for horizontal AntiSaccades (with a 99% CI lower bound of 4.73) and 4.34 from 6 for vertical AntiSaccades (with a 99% CI lower bound of 3.35), are also in line with the performance metrics reported by Antoniades et al. (2013), further validating the DETS system's effectiveness in assessing cognitive control during AntiSaccades tasks.[184]

The qualitative results, particularly those presented in the "Patient Cases" section, highlight the clinical performance of DETS. For instance, in the case of a 33-year-old female with Myasthenia Gravis, DETS detected intra-saccadic fatigue, a finding that substantiates the identification of Pseudo-INO rather than true bilateral INO. These qualitative assessments not only affirm the diagnostic validity of DETS but also demonstrate its capability to elucidate intricate patterns of oculomotor abnormalities unique to specific neurological conditions.

The development of DETS protocols and biometrics were suited to capture a broad spectrum of neurological disorders through a comprehensive assessment of various aspects of ocular motor function. Each protocol's relevance is intrinsically linked to the neural pathways involved in various neurological conditions. For instance, disorders such as Progressive Supranuclear Palsy (PSP), Parkinson's Disease (PD), and certain cerebellar disorders often present with fixation instability, a consequence of disruptions in the brainstem, cerebellum, and basal ganglia. These disruptions can be quantitatively assessed using biometrics such as accuracy, variation, and congruency, while qualitative observations can reveal specific patterns, like direction-changing nystagmus.[48]

In conditions like Multiple System Atrophy (MSA), mass lesions, and strokes, biometrics such as increased saccadic latency, reduced velocity, increased gain, and dysmetria are informative both for localizing the dysfunction to the frontal eye field, superior colliculus, or cerebellum, and for differential diagnosis, as demonstrated in differentiating MSA from Parkinson’s disease.[37] Additionally, disorders such as Spinocerebellar Ataxias (SCA), Huntington’s Disease, and PD often exhibit impaired smooth pursuit with reduced pursuit gain and increased catch-up saccades, indicating dysfunction in the parieto-occipital regions, cerebellum, or basal ganglia. These biometrics can also aid in the early diagnosis, as studies have shown that pursuit abnormalities may precede motor symptoms in early-stage Huntington’s Disease.[185] Similarly, early diagnosis of neurodegenerative disorders such as Alzheimer’s Disease and Frontotemporal Dementia has been assessed by analyzing prosaccades and antisaccades biomarkers, such as correct rate and latency.[186]

Limitations and Future work

The current DETS software, while effective for rapid prototyping and leveraging existing expertise, requires further refinement. The choice of VBA programing language necessitated the reliance on vendor software (Tobii Pro Lab) for stimuli presentation and data recording. This dependency limited the application's full control over the eye-tracking hardware. To overcome this, future development should consider using more robust coding platforms, such as Python or C++, which would allow for direct communication with eye-tracking hardware. This change would enable the use of dynamic stimuli instead of pre-made videos, thus allowing more flexibility in protocols and accurate event timing.

The screen size of eye-tracker displays limit the maximum measurable angles for peripheral vision, particularly in the vertical axis. This physical constraint may lead to incomplete or less accurate assessments of vertical gaze impairments. This limitation could be overcome by integrating Virtual Reality (VR) goggles into the system. VR technology offers a more immersive and controlled environment for eye-tracking studies, allowing for the presentation of dynamic stimuli without the constraints of a fixed screen. VR goggles can also track head movements in real-time, providing more flexibility to introduce Vestibulocochlear reflex protocols and more accurate data, particularly in tasks that involve peripheral vision or head movements.

Since the primary objective of this project was to develop and validate the DETS application, the pilot study's sample size was relatively small which limited stratification of results across different demographic groups. This limitation is particularly significant in generating normative data, which is crucial for clinical diagnostics. Future research should prioritize expanding the sample size and including participants from various demographic and age groups. By including a broader range of participants, we can better understand how demographic factors such as age, gender, and ethnicity may influence eye movement characteristics, leading to more accurate and personalized diagnostic criteria.

While the current study has shown the potential of DETS in identifying specific neurological conditions, further research is needed to evaluate its sensitivity and specificity across a wider range of disorders. Future work should focus on systematically testing DETS in patients with various neurological conditions, including those with overlapping symptoms, to determine how well the tool can differentiate between them. Such research should aim to identify condition-specific eye movement patterns that can serve as biomarkers for early diagnosis, disease progression monitoring, and treatment response evaluation.

The diagnostic efficacy of eye-tracking technology is inherently limited by the physical abilities and cognitive capacities of the patients. Cognitive disorders often affect a patient's ability to perform eye-tracking tasks. Although studies have demonstrated that visual and cognitive impairments significantly impact task performance in eye-tracking, these criteria varied across different research settings.[187] To enhance the robustness of future studies, it would be beneficial to standardize the cognitive inclusion criteria and possibly develop alternative protocols tailored to patients with severe cognitive impairments.

Finally, future work should prioritize collaborations with other research groups and institutions for larger-scale, multicenter studies. These collaborations would allow for the collection of more extensive and diverse data, contributing to the robustness of the normative datasets and the validation of DETS across different clinical environments. Multicenter studies could also facilitate the standardization of DETS protocols and ensure that the tool is adaptable to various healthcare settings, including those with limited resources. By working together, research teams can accelerate the development of DETS and its integration into routine neurological assessments, ultimately improving patient outcomes on a broader scale.

Conclusion

The project "Utilization of Automated Eye-Tracking as an Ancillary Diagnostic Test in Neurological Disorders" set out to address the need for a standardized, scalable, and automated diagnostic tool in clinical neurology by leveraging the potential of eye-tracking technology. The project successfully developed and implemented a comprehensive Diagnostic Eye-Tracking Study (DETS) designed specifically for clinical neurology. This included the creation of a battery of eye-tracking protocols aimed at evaluating both reflexive and voluntary eye movements, such as fixation, saccades, smooth pursuit, optokinetic nystagmus, prosaccades, and antisaccades. The protocols were designed with clinically relevant stimuli, ensuring that the visual tasks presented to patients were appropriate for assessing neurological function. Key biomarkers pertinent to different neurological conditions were identified and defined, providing a robust framework for analyzing eye movement data. These biomarkers included saccadic latency, saccadic gain, pursuit gain, and fixation stability, among others.

The DETS software was developed to automate the entire process, from data acquisition to analysis, incorporating advanced signal processing algorithms to ensure accurate and reliable results, and presenting the result both qualitatively and quantitatively. The software was validated through a pilot study conducted on a small cohort of participants, including both healthy individuals and patients with neurological disorders. The results of the pilot study confirmed the efficacy and reliability of the DETS protocols and the analytical software. The study demonstrated that DETS could successfully detect subtle eye movement abnormalities. The case studies included in the thesis illustrated how DETS could enhance diagnostic accuracy across a range of neurological disorders and provide additional insights into the underlying neural mechanisms of these disorders. The project successfully achieved all its aims and objectives, establishing a strong foundation for future optimizations that have the potential to significantly improve patient outcomes.

References

1. Leigh RJ, Zee DS. The Neurology of Eye Movements. *The Neurology of Eye Movements*. Published online June 2015. doi:10.1093/MED/9780199969289.001.0001

2. Das J, Graham L, Morris R, et al. Eye Movement in Neurological Disorders. *Neuromethods*. 2022;183:185-205. doi:10.1007/978-1-0716-2391-6_11

3. Ansons AM, Davis H. *Diagnosis and Management of Ocular Motility Disorders: Fourth Edition*.; 2013. doi:10.1002/9781118712368

4. Kondziella D, Waldemar G. Neurological Bedside Examination: "Can I Confirm My Anatomical Hypothesis?" In: *Neurology at the Bedside*. ; 2023. doi:10.1007/978-3-031-43335-1_3

5. Strupp ML, Straumann D, Helmchen C. Central Ocular Motor Disorders: Clinical and Topographic Anatomical Diagnosis, Syndromes and Underlying Diseases. *Klin Monbl Augenheilkd*. 2021;238(11). doi:10.1055/a-1654-0632

6. Bedell HE, Stevenson SB. Eye movement testing in clinical examination. *Vision Res*. 2013;90. doi:10.1016/j.visres.2013.02.001

7. Proceedings - ETRA 2023: ACM Symposium on Eye Tracking Research and Applications. *Eye Tracking Research and Applications Symposium (ETRA)*. Published online 2023.

8. Brunyé TT, Drew T, Weaver DL, Elmore JG. A review of eye tracking for understanding and improving diagnostic interpretation. *Cogn Res Princ Implic*. 2019;4(1). doi:10.1186/s41235-019-0159-2

9. Waldthaler J, Stock L, Student J, Sommerkorn J, Dowiasch S, Timmermann L. Antisaccades in Parkinson's Disease: A Meta-Analysis. *Neuropsychol Rev*. 2021;31(4). doi:10.1007/s11065-021-09489-1

10. Brien DC, Riek HC, Yep R, et al. Classification and staging of Parkinson's disease using video-based eye tracking. *Parkinsonism Relat Disord*. 2023;110. doi:10.1016/j.parkreldis.2023.105316

11. Tsitsi P, Benfatto MN, Seimyr GÖ, Larsson O, Svenningsson P, Markaki I. Fixation Duration and Pupil Size as Diagnostic Tools in Parkinson's Disease. *J Parkinsons Dis*. 2021;11(2). doi:10.3233/JPD-202427

12. de Villers-Sidani É, Voss P, Guitton D, Cisneros-Franco JM, Koch NA, Ducharme S. A novel tablet-based software for the acquisition and analysis of gaze and eye movement parameters: a preliminary validation study in Parkinson's disease. *Front Neurol*. 2023;14. doi:10.3389/fneur.2023.1204733

13. Guo L, Normando EM, Shah PA, De Groef L, Cordeiro MF. Oculo-visual abnormalities in Parkinson's disease: Possible value as biomarkers. *Movement Disorders*. 2018;33(9). doi:10.1002/mds.27454

14. Jung I, Kim JS. Abnormal Eye Movements in Parkinsonism and Movement Disorders. *J Mov Disord*. 2019;12(1). doi:10.14802/jmd.18034/J

15. Wong OWH, Fung GPC, Chan S. Characterizing the relationship between eye movement parameters and cognitive functions in non-demented parkinson's disease patients with eye tracking. *Journal of Visualized Experiments*. 2019;2019(151). doi:10.3791/60052

16. Lisong W, Jiaqi S, Yang L. A logistic regression model for diagnosing Alzheimer ' s disease based on eye movement tracking technology. *Journal of Army Medical University*. 2023;45(2). doi:10.16016/j.2097-0927.202207007

17. Zuo F, Jing P, Sun J, Duan J, Ji Y, Liu Y. Deep Learning-Based Eye-Tracking Analysis for Diagnosis of Alzheimer's Disease Using 3D Comprehensive Visual Stimuli. *IEEE J Biomed Health Inform*. 2024;28(5). doi:10.1109/JBHI.2024.3365172

18. Chehrehnegar N, Shati M, Esmaeili M, Foroughan M. Executive function deficits in mild cognitive impairment: evidence from saccade tasks. *Aging Ment Health*. 2022;26(5). doi:10.1080/13607863.2021.1913471

19. Ionescu A, Ștefănescu E, Strilciuc Ștefan, Rafila A, Mureșanu D. Correlating Eye-Tracking Fixation Metrics and Neuropsychological Assessment after Ischemic Stroke. *Medicina (Lithuania)*. 2023;59(8). doi:10.3390/medicina59081361

20. Abul Hassan M, Aldridge CM, Zhuang Y, et al. Approach to Quantify Eye Movements to Augment Stroke Diagnosis With a Non-Calibrated Eye-Tracker. *IEEE Trans Biomed Eng*. 2023;70(6):1750-1757. doi:10.1109/TBME.2022.3227015

21. Lopergolo D, Rosini F, Pretegiani E, Bargagli A, Serchi V, Rufa A. Autosomal recessive cerebellar ataxias: a diagnostic classification approach according to ocular features. *Front Integr Neurosci*. 2023;17. doi:10.3389/fnint.2023.1275794

22. Szpisjak L, Szaraz G, Salamon A, et al. Eye-tracking-aided characterization of saccades and antisaccades in SYNE1 ataxia patients: a pilot study. *BMC Neurosci*. 2021;22(1). doi:10.1186/s12868-021-00612-9

23. Wolf A, Tripanpitak K, Umeda S, Otake-Matsuura M. Eye-tracking paradigms for the assessment of mild cognitive impairment: a systematic review. *Front Psychol*. 2023;14. doi:10.3389/fpsyg.2023.1197567

24. Kritika Singh. 14 Best Eye Tracking Software for Marketing and Online Research. Geekflare.

25. Tobii AB. *Tobii Pro Lab User Manual*.; 2024.

26. Tahri Sqalli M, Aslonov B, Gafurov M, Mukhammadiev N, Sqalli Houssaini Y. Eye tracking technology in medical practice: a perspective on its diverse applications. *Front Med Technol*. 2023;5. doi:10.3389/fmedt.2023.1253001

27. Li W, Zhou W, Fei M, Xu Y, Yang E. Eye Tracking Methodology for Diagnosing Neurological Diseases: A Survey. In: *Proceedings - 2020 Chinese Automation Congress, CAC 2020*. ; 2020. doi:10.1109/CAC51589.2020.9326691

28. Adhanom IB, MacNeilage P, Folmer E. Eye Tracking in Virtual Reality: a Broad Review of Applications and Challenges. *Virtual Real*. 2023;27(2). doi:10.1007/s10055-022-00738-z

29. Oyama A, Takeda S, Ito Y, et al. Novel Method for Rapid Assessment of Cognitive Impairment Using High-Performance Eye-Tracking Technology. *Sci Rep*. 2019;9(1). doi:10.1038/s41598-019-49275-x

30. Ke F, Liu R, Sokolikj Z, Dahlstrom-Hakki I, Israel M. Using eye-tracking in education: review of empirical research and technology. *Educational Technology Research and Development*. Published online 2024. doi:10.1007/s11423-024-10342-4

31. Klaib AF, Alsrehin NO, Melhem WY, Bashtawi HO, Magableh AA. Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies. *Expert Syst Appl*. 2021;166. doi:10.1016/j.eswa.2020.114037

32. Rodríguez-Labrada R, Vázquez-Mojena Y, Velázquez-Pérez L. Eye Movement Abnormalities in Neurodegenerative Diseases. In: *Eye Motility*. ; 2019. doi:10.5772/intechopen.81948

33. Sekar A, Panouillères MTN, Kaski D. Detecting Abnormal Eye Movements in Patients with Neurodegenerative Diseases – Current Insights. *Eye Brain*. 2024;16:3-16. doi:10.2147/EB.S384769

34. Anderson TJ, MacAskill MR. Eye movements in patients with neurodegenerative disorders. *Nat Rev Neurol*. 2013;9(2):74-85. doi:10.1038/nrneurol.2012.273

35. Grimm MJ, Respondek G, Stamelou M, et al. Clinical Conditions "Suggestive of Progressive Supranuclear Palsy"—Diagnostic Performance. *Movement Disorders*. 2020;35(12). doi:10.1002/mds.28263

36. Peltsch A, Hemraj A, Garcia A, Munoz DP. Saccade deficits in amnestic mild cognitive impairment resemble mild Alzheimer's disease. *Eur J Neurosci*. 2014;39(11):2000-2013. doi:10.1111/EJN.12617

37. Krismer F, Wenning GK. Multiple system atrophy: Insights into a rare and debilitating movement disorder. *Nat Rev Neurol*. 2017;13(4). doi:10.1038/nrneurol.2017.26

38. Polden M, Crawford TJ. Eye Movement Latency Coefficient of Variation as a Predictor of Cognitive Impairment: An Eye Tracking Study of Cognitive Impairment. *Vision (Switzerland)*. 2023;7(2). doi:10.3390/vision7020038

39. Opwonya J, Wang C, Jang KM, Lee K, Kim J Il, Kim JU. Inhibitory Control of Saccadic Eye Movements and Cognitive Impairment in Mild Cognitive Impairment. *Front Aging Neurosci*. 2022;14. doi:10.3389/fnagi.2022.871432

40. Armstrong MJ, Litvan I, Lang AE, et al. Criteria for the diagnosis of corticobasal degeneration. *Neurology*. 2013;80(5). doi:10.1212/WNL.0b013e31827f0fd1

41. Shields M, Sinkar S, Chan WO, Crompton J. Parinaud syndrome: a 25-year (1991–2016) review of 40 consecutive adult cases. *Acta Ophthalmol*. 2017;95(8). doi:10.1111/aos.13283

42. Ohba N, Ohba A, Sato N. Parinaud syndrome. *Neuro-Ophthalmology Japan*. 2018;35(1). doi:10.11476/shinkeiganka.35.89

43. Choi JY, Lee SH, Kim JS. Central vertigo. *Curr Opin Neurol*. 2018;31(1). doi:10.1097/WCO.0000000000000511

44. Fattal D, Platti N. Ocular Lateral Deviation as a Vestibular Sign to Improve Detection of Posterior Circulation Strokes: A Review of the Literature. *Journal of Emergency Medicine*. 2023;64(5). doi:10.1016/j.jemermed.2023.02.010

45. Garces P, Antoniades CA, Sobanska A, et al. Quantitative Oculomotor Assessment in Hereditary Ataxia: Discriminatory Power, Correlation with Severity Measures, and Recommended Parameters for Specific Genotypes. *Cerebellum*. 2024;23(1). doi:10.1007/s12311-023-01514-8

46. Lestari DT, Hidayati HB. Acute Vestibular Syndrome in Cerebellar Infarction: A Case Report. *International Journal of Research and Review*. 2021;8(9). doi:10.52403/ijrr.20210906

47. Kassavetis P, Kaski D, Anderson T, Hallett M. Eye Movement Disorders in Movement Disorders. *Mov Disord Clin Pract*. 2022;9(3):284-295. doi:10.1002/mdc3.13413

48. Lal V, Truong D. Eye movement abnormalities in movement disorders. *Clin Park Relat Disord*. 2019;1:54-63. doi:10.1016/j.prdoa.2019.08.004

49. Arnulf I, Nielsen J, Lohmann E, et al. Rapid eye movement sleep disturbances in Huntington disease. *Arch Neurol*. 2008;65(4). doi:10.1001/archneur.65.4.482

50. Grabska N, Rudzińska M, Wójcik-Pędziwiatr M, Michalski M, Sławek J, Szczudlik A. Saccadic eye movements in juvenile variant of Huntington disease. *Neurol Neurochir Pol*. 2014;48(4). doi:10.1016/j.pjnns.2014.06.003

51. Ortiz JF, Morillo Cox Á, Tambo W, et al. Neurological Manifestations of Wilson's Disease: Pathophysiology and Localization of Each Component. *Cureus*. Published online 2020. doi:10.7759/cureus.11509

52. Berman RA, Colby CL, Genovese CR, et al. Cortical networks subserving pursuit and saccadic eye movements in humans: An FMRI study. *Hum Brain Mapp*. 1999;8(4). doi:10.1002/(SICI)1097-0193(1999)8:4<209::AID-HBM5>3.0.CO;2-0

53. Schoppik D, Nagel KI, Lisberger SG. Cortical Mechanisms of Smooth Eye Movements Revealed by Dynamic Covariations of Neural and Behavioral Responses. *Neuron*. 2008;58(2). doi:10.1016/j.neuron.2008.02.015

54. Liu B, MacEllaio M V., Osborne LC. Efficient sensory cortical coding optimizes pursuit eye movements. *Nat Commun*. 2016;7. doi:10.1038/ncomms12759

55. Pierrot-Deseilligny C, Rivaud S, Gaymard B, Müri R, Vermersch A -I. Cortical control of saccades. *Ann Neurol*. 1995;37(5). doi:10.1002/ana.410370504

56. Gaymard B, Ploner CJ, Rivaud S, Vermersch AI, Pierrot-Deseilligny C. Cortical control of saccades. In: *Experimental Brain Research*. Vol 123. ; 1998. doi:10.1007/s002210050557

57. Nij Bijvank JA, Hof SN, Prouskas SE, et al. A novel eye-movement impairment in multiple sclerosis indicating widespread cortical damage. *Brain*. 2023;146(6). doi:10.1093/brain/awac474

58. Lee SU, Kim HJ, Choi JH, Choi JY, Kim JS. Comparison of Ocular Motor Findings Between Neuromyelitis Optica Spectrum Disorder and Multiple Sclerosis Involving the Brainstem and Cerebellum. *Cerebellum*. 2019;18(3). doi:10.1007/s12311-019-01018-4

59. Costa Novo J, Felgueiras H. Neuro-ophthalmologic manifestations of multiple sclerosis other than acute optic neuritis. *Mult Scler Relat Disord*. 2021;48. doi:10.1016/j.msard.2020.102730

60. Stojkovic T, Béhin A. Ocular myasthenia: Diagnosis and treatment. *Rev Neurol (Paris)*. 2010;166(12). doi:10.1016/j.neurol.2010.08.004

61. Chao YS, Argáez C. Video-Oculography for the Diagnosis of Ocular Myasthenia Gravis: A Review of Diagnostic Accuracy, Cost-Effectiveness, and Guidelines. *Video-Oculography for the Diagnosis of Ocular Myasthenia Gravis: A Review of Diagnostic Accuracy, Cost-Effectiveness, and Guidelines*. Published online 2019.

62. Corrigendum: A patient with korsakoff syndrome of psychiatric and alcoholic etiology presenting as DSM-5 mild neurocognitive disorder (T. Neuropsychiatr Dis Treat., (2019) 15, (1311-1320), 10.2147/NDT.S203513). *Neuropsychiatr Dis Treat*. 2019;15. doi:10.2147/NDT.S220735

63. Sechi G Pietro, Serra A. Wernicke's encephalopathy: new clinical settings and recent advances in diagnosis and management. *Lancet Neurology*. 2007;6(5). doi:10.1016/S1474-4422(07)70104-7

64. Hickman S. Paraneoplastic syndromes in neuro-ophthalmology. *Ann Indian Acad Neurol*. 2022;25(8). doi:10.4103/aian.aian_102_22

65. Keime-Guibert F, Graus F, Broët P, et al. Clinical outcome of patients with anti-Hu-associated encephalomyelitis after treatment of the tumor. *Neurology*. 1999;53(8). doi:10.1212/wnl.53.8.1719

66. Letendre SL, Ellis RJ, Ances BM, McCutchan JA. Neurologic complications of HIV disease and their treatment. *Top HIV Med*. 2010;18(2).

67. Kolson D. Neurologic complications of HIV infection in the era of antiretroviral therapy. *Top Antivir Med*. 2017;25(3).

68. Bhatia NS, Chow FC. Neurologic Complications in Treated HIV-1 Infection. *Curr Neurol Neurosci Rep*. 2016;16(7). doi:10.1007/s11910-016-0666-1

69. Richards PM. Mild traumatic brain injury and postconcussion syndrome: The new evidence base for diagnosis and treatment. *Psychol Inj Law*. 2009;2(1). doi:10.1007/s12207-009-9036-5

70. Maruta J, Jaw E, Modera P, Rajashekar U, Spielman LA, Ghajar J. Frequency responses to visual tracking stimuli may be affected by concussion. *Mil Med*. 2017;182. doi:10.7205/MILMED-D-16-00093

71. Martinez-Conde S, Macknik SL, Hubel DH. The role of fixational eye movements in visual perception. *Nat Rev Neurosci*. 2004;5(3). doi:10.1038/nrn1348

72. Van Opstal AJ, Goossens HHLM. Linear ensemble-coding in midbrain superior colliculus specifies the saccade kinematics. *Biol Cybern*. 2008;98(6). doi:10.1007/s00422-008-0219-z

73. Carpenter RHS. Contrast, probability, and saccadic latency: Evidence for independence of detection and decision. *Current Biology*. 2004;14(17). doi:10.1016/j.cub.2004.08.058

74. Carl JR, Gellman RS. Human smooth pursuit: Stimulus-dependent responses. *J Neurophysiol*. 1987;57(5). doi:10.1152/jn.1987.57.5.1446

75. Krauzlis RJ, Stone LS. Tracking with the mind's eye. *Trends Neurosci*. 1999;22(12). doi:10.1016/S0166-2236(99)01464-2

76. Barnes GR. Cognitive processes involved in smooth pursuit eye movements. *Brain Cogn*. 2008;68(3). doi:10.1016/j.bandc.2008.08.020

77. Lisberger SG. Visual Guidance of Smooth Pursuit Eye Movements. *Annu Rev Vis Sci*. 2015;1(1). doi:10.1146/annurev-vision-082114-035349

78. van Loon AM, Olmos-Solis K, Olivers CNL. Subtle eye movement metrics reveal task-relevant representations prior to visual search. *J Vis*. 2017;17(6). doi:10.1167/17.6.13

79. Mays LE, Gamlin PD. Neuronal circuitry controlling the near response. *Curr Opin Neurobiol*. 1995;5(6). doi:10.1016/0959-4388(95)80104-9

80. Huterer M, Cullen KE. Vestibuloocular reflex dynamics during high-frequency and high-acceleration rotations of the head on body in Rhesus monkey. *J Neurophysiol*. 2002;88(1). doi:10.1152/jn.2002.88.1.13

81. Crane BT, Demer JL. Human horizontal vestibulo-ocular reflex initiation: Effects of acceleration, target distance, and unilateral deafferentation. *J Neurophysiol*. 1998;80(3). doi:10.1152/jn.1998.80.3.1151

82. Worfolk R, Barnes GR. Interaction of active and passive slow eye movement systems. *Exp Brain Res*. 1992;90(3):589-598. doi:10.1007/BF00230943

83. Ilg UJ. Slow eye movements. *Prog Neurobiol*. 1997;53(3):293-329. doi:https://doi.org/10.1016/S0301-0082(97)00039-7

84. Cohen B, Matsuo V, Raphan T. Quantitative analysis of the velocity characteristics of optokinetic nystagmus and optokinetic after-nystagmus. *J Physiol*. 1977;270(2). doi:10.1113/jphysiol.1977.sp011955

85. Martinez-Conde S, Otero-Millan J, MacKnik SL. The impact of microsaccades on vision: Towards a unified theory of saccadic function. *Nat Rev Neurosci*. 2013;14(2). doi:10.1038/nrn3405

86. Rolfs M. Microsaccades: Small steps on a long way. *Vision Res*. 2009;49(20). doi:10.1016/j.visres.2009.08.010

87. Hafed ZM, Clark JJ. Microsaccades as an overt measure of covert attention shifts. *Vision Res*. 2002;42(22). doi:10.1016/S0042-6989(02)00263-8

88. Otero-Millan J, Macknik SL, Martinez-Conde S. Fixational eye movements and binocular vision. *Front Integr Neurosci*. 2014;8(JUL). doi:10.3389/fnint.2014.00052

89. Bowers NR, Boehm AE, Roorda A. The effects of fixational tremor on the retinal image. *J Vis*. 2019;19(11). doi:10.1167/19.11.8

90. Rucci M, Iovin R, Poletti M, Santini F. Miniature eye movements enhance fine spatial detail. *Nature*. 2007;447(7146). doi:10.1038/nature05866

91. Jacobs GH. Primate color vision: A comparative perspective. *Vis Neurosci*. 2008;25(5-6). doi:10.1017/S0952523808080760

92. Martinez-Conde S, Macknik SL. Fixational eye movements across vertebrates: Comparative dynamics, physiology, and perception. *J Vis*. 2008;8(14). doi:10.1167/8.14.28

93. Rucci M, Poletti M. Control and Functions of Fixational Eye Movements. *Annu Rev Vis Sci*. 2015;1. doi:10.1146/annurev-vision-082114-035742

94. Moschovakis AK, Scudder CA, Highstein SM. The microscopic anatomy and physiology of the mammalian saccadic system. *Prog Neurobiol*. 1996;50(2-3). doi:10.1016/S0301-0082(96)00034-2

95. Haines DE, Mihailoff GA. *Fundamental Neuroscience for Basic and Clinical Applications: Fifth Edition*.; 2017.

96. Tarnutzer AA, Straumann D. Nystagmus. *Curr Opin Neurol*. 2018;31(1). https://journals.lww.com/co-neurology/fulltext/2018/02000/nystagmus.12.aspx

97. Bede P, Finegan E, Chipika RH, et al. Occulomotor neural integrator dysfunction in multiple sclerosis: Insights from neuroimaging. *Front Neurol*. 2018;9(AUG). doi:10.3389/fneur.2018.00691

98. Tilikete C, Pélisson D. Ocular motor syndromes of the brainstem and cerebellum. *Curr Opin Neurol*. 2008;21(1). doi:10.1097/WCO.0b013e3282f4097d

99. Matsuyoshi H, Yamanishi T, Goto H, Miwa T, Kurisaki R. Clinical analysis of 17 cases with down beat nystagmus. *Otolaryngology - Head and Neck Surgery (Japan)*. 2016;88(7).

100. Ling X, Wu YX, Feng YF, et al. Spontaneous nystagmus with an upbeat component: Central or peripheral vestibular disorders? *Front Neurol*. 2023;14. doi:10.3389/fneur.2023.1106084

101. Adams OE, Olson SB, Lam H, et al. Complete and Immediate Resolution of See-Saw Nystagmus Following Pituitary Macroadenoma Resection: Case Report and Review of the Literature. *Neuro-Ophthalmology*. Published online 2024. doi:10.1080/01658107.2023.2299763

102. Thier P, Markanday A. Role of the Vermal Cerebellum in Visually Guided Eye Movements and Visual Motion Perception. *Annu Rev Vis Sci*. 2019;5. doi:10.1146/annurev-vision-091718-015000

103. Giolli RA, Blanks RHI, Lui F. The accessory optic system: Basic organization with an update on connectivity, neurochemistry, and function. *Prog Brain Res*. 2006;151. doi:10.1016/S0079-6123(05)51013-6

104. Sun LO, Brady CM, Cahill H, et al. Functional Assembly of Accessory Optic System Circuitry Critical for Compensatory Eye Movements. *Neuron*. 2015;86(4). doi:10.1016/j.neuron.2015.03.064

105. Yetiser S, Ince D, Yetiser B. Optokinetic Analysis in Patients With Spontaneous Horizontal Gaze-Evoked Nystagmus Without Radiological Neuropathology. *Ear Nose Throat J*. 2019;98(7). doi:10.1177/0145561319840902

106. Helmchen C, Heide W, Strupp M, Straumann D. Update on central oculomotor disorders and nystagmus. *Nervenheilkunde*. 2023;42(1-2). doi:10.1055/a-1946-6812

107. Baier B, Stoeter P, Dieterich M. Anatomical correlates of ocular motor deficits in cerebellar lesions. *Brain*. 2009;132(8). doi:10.1093/brain/awp165

108. Gamlin PDR. Neural mechanisms for the control of vergence eye movements. In: *Annals of the New York Academy of Sciences*. Vol 956. ; 2002. doi:10.1111/j.1749-6632.2002.tb02825.x

109. Gibaldi A, Sabatini SP. The saccade main sequence revised: A fast and repeatable tool for oculomotor analysis. *Behav Res Methods*. 2021;53(1). doi:10.3758/s13428-020-01388-2

110. Termsarasab P, Thammongkolchai T, Rucker JC, Frucht SJ. The diagnostic value of saccades in movement disorder patients: a practical guide and review. *J Clin Mov Disord*. 2015;2(1). doi:10.1186/s40734-015-0025-4

111. Sparks DL. The brainstem control of saccadic eye movements. *Nat Rev Neurosci*. 2002;3(12). doi:10.1038/nrn986

112. Crapse TB, Sommer MA. Frontal eye field neurons with spatial representations predicted by their subcortical input. *Journal of Neuroscience*. 2009;29(16). doi:10.1523/JNEUROSCI.4906-08.2009

113. Bedini M, Olivetti E, Avesani P, Baldauf D. Accurate localization and coactivation profiles of the frontal eye field and inferior frontal junction: an ALE and MACM fMRI meta-analysis. *Brain Struct Funct*. 2023;228(3-4). doi:10.1007/s00429-023-02641-y

114. Matsumoto M, Inoue KI, Takada M. Causal Role of Neural Signals Transmitted From the Frontal Eye Field to the Superior Colliculus in Saccade Generation. *Front Neural Circuits*. 2018;12. doi:10.3389/fncir.2018.00069

115. Jia J, Puyang Z, Wang Q, Jin X, Chen A. Dynamic encoding of saccade sequences in primate frontal eye field. *Journal of Physiology*. 2021;599(22). doi:10.1113/JP282094

116. Pouget P, Stepniewska I, Crowder EA, et al. Visual and motor connectivity and the distribution of calcium-binding proteins in macaque frontal eye field: Implications for saccade target selection. *Front Neuroanat*. 2009;3(MAY). doi:10.3389/neuro.05.002.2009

117. Sparks D, Rohrer WH, Zhang Y. The role of the superior colliculus in saccade initiation: A study of express saccades and the gap effect. *Vision Res*. 2000;40(20). doi:10.1016/S0042-6989(00)00133-4

118. Vernet M, Quentin R, Chanes L, Mitsumasu A, Valero-Cabré A. Frontal eye field, where art thou? Anatomy, function, and non-invasive manipulation of frontal regions involved in eye movements and associated cognitive operations. *Front Integr Neurosci*. 2014;8(AUG). doi:10.3389/fnint.2014.00066

119. Terao Y, Fukuda H, Yugeta A, et al. Initiation and inhibitory control of saccades with the progression of Parkinson's disease - Changes in three major drives converging on the superior colliculus. *Neuropsychologia*. 2011;49(7). doi:10.1016/j.neuropsychologia.2011.03.002

120. Hikosaka O, Isoda M. Switching from automatic to controlled behavior: cortico-basal ganglia mechanisms. *Trends Cogn Sci*. 2010;14(4). doi:10.1016/j.tics.2010.01.006

121. Bhidayasiri R, Somers JT, Kim JI, et al. Ocular oscillations induced by shifts of the direction and depth of visual fixation. *Ann Neurol*. 2001;49(1). doi:10.1002/1531-8249(200101)49:1<24::AID-ANA6>3.0.CO;2-T

122. Barnes GR. Ocular pursuit movements. In: *The Oxford Handbook of Eye Movements*. ; 2012. doi:10.1093/oxfordhb/9780199539789.013.0007

123. Kang JJ, Lee SU, Kim JM, Oh SY. Recording and interpretation of ocular movements: saccades, smooth pursuit, and optokinetic nystagmus. *Annals of Clinical Neurophysiology*. 2023;25(2). doi:10.14253/acn.2023.25.2.55

124. Himmelberg MM, Winawer J, Carrasco M. Linking individual differences in human primary visual cortex to contrast sensitivity around the visual field. *Nat Commun*. 2022;13(1). doi:10.1038/s41467-022-31041-9

125. Schütz AC, Braun DI, Gegenfurtner KR. Eye movements and perception: A selective review. *J Vis*. 2011;11(5). doi:10.1167/11.5.1

126. Schutz AC, Braun DI, Gegenfurtner KR. Eye movements and perception: A selective review. *J Vis*. 2011;11(5). doi:10.1167/11.5.9

127. Van Essen DC, Gallant JL. Neural mechanisms of form and motion processing in the primate visual system. *Neuron*. 1994;13(1). doi:10.1016/0896-6273(94)90455-3

128. Newsome WT, Pare EB. A selective impairment of motion perception following lesions of the middle temporal visual area (MT). *Journal of Neuroscience*. 1988;8(6). doi:10.1523/jneurosci.08-06-02201.1988

129. Berman RA, Wurtz RH. Signals conveyed in the pulvinar pathway from superior colliculus to cortical area MT. *Journal of Neuroscience*. 2011;31(2). doi:10.1523/JNEUROSCI.4738-10.2011

130. Thier P, Ilg UJ. The neural basis of smooth-pursuit eye movements. *Curr Opin Neurobiol*. 2005;15(6). doi:10.1016/j.conb.2005.10.013

131. Kheradmand A, Zee DS. Cerebellum and ocular motor control. *Front Neurol*. 2011;SEP. doi:10.3389/fneur.2011.00053

132. Schröder R, Keidel K, Trautner P, Radbruch A, Ettinger U. Neural mechanisms of background and velocity effects in smooth pursuit eye movements. *Hum Brain Mapp*. 2023;44(3). doi:10.1002/hbm.26127

133. Merriam EP, Gardner JL, Movshon JA, Heeger DJ. Modulation of visual responses by gaze direction in human visual cortex. *Journal of Neuroscience*. 2013;33(24). doi:10.1523/JNEUROSCI.0500-12.2013

134. Pinkhardt EH, Kassubek J. Ocular motor abnormalities in Parkinsonian syndromes. *Parkinsonism Relat Disord*. 2011;17(4):223-230. doi:10.1016/J.PARKRELDIS.2010.08.004

135. Dash S, Nazari SA, Yan X, Wang H, Crawford JD. Superior colliculus responses to attended, unattended, and remembered saccade targets during smooth pursuit eye movements. *Front Syst Neurosci*. 2016;10(APR). doi:10.3389/fnsys.2016.00034

136. White BJ, Itti L, Munoz DP. Superior colliculus encodes visual saliency during smooth pursuit eye movements. *European Journal of Neuroscience*. 2021;54(1). doi:10.1111/ejn.14432

137. Yokota JI, Ota Y, Yabe T, Shimoda S. The neuro-otological studies of two patients with Opalski's syndrome experiencing vertigo/dizziness. *Equilibrium Research*. 2016;75(1). doi:10.3757/jser.75.7

138. Bjeloš M, Križanović A, Bušić M, Elabjer BK. Visual Deficiency in Wallenberg's Syndrome. *Coll Antropol*. 2021;45(2). doi:10.5671/ca.45.2.4

139. Estrany B, Fuster-Parra P. Human Eye Tracking Through Electro-Oculography (EOG): A Review. In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol 13492 LNCS. ; 2022. doi:10.1007/978-3-031-16538-2_8

140. You JL, Hui YN, Zhang L. Eye movements and progression of clinical applications of eye tracking technology. *International Eye Science*. 2023;23(1). doi:10.3980/j.issn.1672-5123.2023.1.18

141. Raghavendran S, Vadivel KD. Corneal reflection based eye tracking technology to overcome the limitations of infrared rays based eye tracking. In: *AIP Conference Proceedings*. Vol 2831. ; 2023. doi:10.1063/5.0162849

142. Tobii. Tobii Eye Tracking - An introduction to eye tracking and Tobii Eye Trackers. *Technology (Singap World Sci)*. Published online 2010.

143. Tobii. Tobii Pro Spectrum eye trackers. https://www.tobii.com/products/accessories/chin-rest.

144. Shi L, Wang CY, Tian F, Jia HB. An integrated neural network model for pupil detection and tracking. *Soft comput*. 2021;25(15). doi:10.1007/s00500-021-05984-y

145. Gneo M, Schmid M, Conforto S, D'Alessio T. A free geometry model-independent neural eye-gaze tracking system. *J Neuroeng Rehabil*. 2012;9(1). doi:10.1186/1743-0003-9-82

146. Mestre C, Gautier J, Pujol J. Robust eye tracking based on multiple corneal reflections for clinical applications. *J Biomed Opt*. 2018;23(03). doi:10.1117/1.jbo.23.3.035001

147. Tobii. 3D coordinate system. https://connect.tobii.com/s/article/Display-Area-Coordinate-System-DACS?language=en_US.

148. Kennedy A. Book Review: Eye Tracking: A Comprehensive Guide to Methods and Measures. *Quarterly Journal of Experimental Psychology*. 2016;69(3). doi:10.1080/17470218.2015.1098709

149. Duchowski AT. *Eye Tracking Methodology: Theory and Practice: Third Edition*.; 2017. doi:10.1007/978-3-319-57883-5

150. Hassoumi A, Peysakhovich V, Hurter C. Improving eye-tracking calibration accuracy using symbolic regression. *PLoS One*. 2019;14(3). doi:10.1371/journal.pone.0213675

151. Brilhault A, Neuenschwander S, Rios RA. A new robust multivariate mode estimator for eye-tracking calibration. *Behav Res Methods*. 2023;55(2). doi:10.3758/s13428-022-01809-4

152. Griffith H, Lohr D, Abdulin E, Komogortsev O. GazeBase, a large-scale, multi-stimulus, longitudinal eye movement dataset. *Sci Data*. 2021;8(1). doi:10.1038/s41597-021-00959-y

153. Li B, Barney E, Hudac C, et al. Selection of eye-tracking stimuli for prediction by sparsely grouped input variables for neural networks: Towards biomarker refinement for autism. In: *Eye Tracking Research and Applications Symposium (ETRA)*. ; 2020. doi:10.1145/3379155.3391334

154. Godfroid A, Hui B. Five common pitfalls in eye-tracking research. *Second Lang Res*. 2020;36(3). doi:10.1177/0267658320921218

155. Alhilo T, Al-Sakaa A. Handling Noisy Data in Eye-Tracking Research: Methods and Best Practices. In: *2023 International Workshop on Biomedical Applications, Technologies and Sensors (BATS)*. ; 2023:39-44. doi:10.1109/BATS59463.2023.10303090

156. Peng W. EEG preprocessing and denoising. In: *EEG Signal Processing and Feature Extraction*. ; 2019. doi:10.1007/978-981-13-9113-2_5

157. Shan Z, Yang J, Sanjuán MAF, Wu C, Liu H. A novel adaptive moving average method for signal denoising in strong noise background. *Eur Phys J Plus*. 2022;137(1). doi:10.1140/epjp/s13360-021-02279-x

158. Mansour EHA, Bretaudeau F. A novel edge detection method based on efficient gaussian binomial filter. *International Journal of Advances in Intelligent Informatics*. 2021;7(2). doi:10.26555/ijain.v7i2.651

159. Chong ZL, Tan KL, Khoo MBC, Teoh WL, Castagliola P. Optimal designs of the exponentially weighted moving average (EWMA) median chart for known and estimated parameters based on median run length. *Commun Stat Simul Comput*. 2022;51(7). doi:10.1080/03610918.2020.1721539

160. Baldazzi G, Solinas G, Valle J Del, et al. Systematic analysis of wavelet denoising methods for neural signal processing. *J Neural Eng*. 2020;17(6). doi:10.1088/1741-2552/abc741

161. Gallagher NB. Savitzky–Golay Smoothing and Differentitation Filter. *Eigenvector*. Published online 2020.

162. Hessels RS, Niehorster DC, Kemner C, Hooge ITC. Noise-robust fixation detection in eye movement data: Identification by two-means clustering (I2MC). *Behav Res Methods*. 2017;49(5). doi:10.3758/s13428-016-0822-1

163. Ma J, Fan N, Wang N. Normal Visual Field. In: Wang N, Liu X, Fan N, eds. *Optic Disorders and Visual Field*. Springer Singapore; 2019:43-48. doi:10.1007/978-981-13-2502-1_9

164. Hofer H, Carroll J, Neitz J, Neitz M, Williams DR. Organization of the human trichromatic cone mosaic. *Journal of Neuroscience*. 2005;25(42). doi:10.1523/JNEUROSCI.2414-05.2005

165. Mangalam M, Kelty-Stephen DG, Hayano J, Watanabe E, Kiyono K. Quantifying non-Gaussian intermittent fluctuations in physiology: Multiscale probability density function analysis using the Savitzky-Golay detrending. *Phys Rev Res*. 2023;5(4). doi:10.1103/PhysRevResearch.5.043157

166. Ehinger B V., Groß K, Ibs I, König P. A new comprehensive eye-tracking test battery concurrently evaluating the Pupil Labs glasses and the EyeLink 1000. *PeerJ*. 2019;2019(7). doi:10.7717/peerj.7086

167. Niehorster DC, Cornelissen THW, Holmqvist K, Hooge ITC, Hessels RS. What to expect from your remote eye-tracker when participants are unrestrained. *Behav Res Methods*. 2018;50(1). doi:10.3758/s13428-017-0863-0

168. Opwonya J, Ku B, Lee KH, Kim J Il, Kim JU. Eye movement changes as an indicator of mild cognitive impairment. *Front Neurosci*. 2023;17. doi:10.3389/fnins.2023.1171417

169. Leharanger M, Liu P, Vandromme L, Balédent O. Eye Tracking Post Processing to Detect Visual Artifacts and Quantify Visual Attention under Cognitive Task Activity during fMRI. *Sensors*. 2024;24(15). doi:10.3390/s24154916

170. Imaoka Y, Flury A, de Bruin ED. Assessing Saccadic Eye Movements With Head-Mounted Display Virtual Reality Technology. *Front Psychiatry*. 2020;11. doi:10.3389/fpsyt.2020.572938

171. Collewijn H, Erkelens CJ, Steinman RM. Binocular co-ordination of human horizontal saccadic eye movements. *J Physiol*. 1988;404(1). doi:10.1113/jphysiol.1988.sp017284

172. Collewijn H, Erkelens CJ, Steinman RM. Binocular co-ordination of human vertical saccadic eye movements. *J Physiol*. 1988;404(1). doi:10.1113/jphysiol.1988.sp017285

173. Boghen D, Troost BT, Daroff RB, Dell'Osso LF, Birkett JE. Velocity characteristics of normal human saccades. *Invest Ophthalmol*. 1974;13(8).

174. Guadron L, van Opstal AJ, Goossens J. Speed-accuracy tradeoffs influence the main sequence of saccadic eye movements. *Sci Rep*. 2022;12(1). doi:10.1038/s41598-022-09029-8

175. Lima D da S, Ventura DF. A review of experimental task design in psychophysical eye tracking research. *Front Hum Neurosci*. 2023;17. doi:10.3389/fnhum.2023.1112769

176. Lencer R, Sprenger A, Trillenberg P. Smooth Eye Movements in Humans: Smooth Pursuit, Optokinetic Nystagmus and Vestibular Ocular Reflex. In: ; 2019. doi:10.1007/978-3-030-20085-5_4

177. Lemos J, Eggenberger E. Saccadic intrusions: Review and update. *Curr Opin Neurol*. 2013;26(1):59-66. doi:10.1097/WCO.0b013e32835c5e1d

178. Norouzifard M, Black J, Thompson B, Klette R, Turuwhenua J. A Real-Time Eye Tracking Method for Detecting Optokinetic Nystagmus. In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol 12047 LNCS. ; 2020. doi:10.1007/978-3-030-41299-9_12

179. Hunfalvay M, Roberts CM, Murray N, Tyagi A, Kelly H, Bolte T. Horizontal and vertical self-paced saccades as a diagnostic marker of traumatic brain injury. *Concussion*. 2019;4(1). doi:10.2217/cnc-2019-0001

180. Irving EL, Lillakas L. Difference between vertical and horizontal saccades across the human lifespan. *Exp Eye Res*. 2019;183. doi:10.1016/j.exer.2018.08.020

181. Ettinger U, Ffytche DH, Kumari V, et al. Decomposing the neural correlates of antisaccade eye movements using event-related fmri. *Cerebral Cortex*. 2008;18(5). doi:10.1093/cercor/bhm147

182. Hutton SB. Cognitive control of saccadic eye movements. *Brain Cogn*. 2008;68(3):327-340. doi:10.1016/j.bandc.2008.08.021

183. Munoz DP, Everling S. Look away: The anti-saccade task and the voluntary control of eye movement. *Nat Rev Neurosci*. 2004;5(3). doi:10.1038/nrn1345

184. Antoniades C, Ettinger U, Gaymard B, et al. An internationally standardised antisaccade protocol. *Vision Res*. 2013;84. doi:10.1016/j.visres.2013.02.007

185. Hicks SL, P.A. Robert M, V.P. Golding C, Tabrizi SJ, Kennard C. Oculomotor deficits indicate the progression of Huntington's Disease. In: *Progress in Brain Research*. Vol 171. ; 2008. doi:10.1016/S0079-6123(08)00678-X

186. Zhang S, Huang X, An R, Xiao W, Wan Q. The application of saccades to assess cognitive impairment among older adults: a systematic review and meta-analysis. *Aging Clin Exp Res*. 2023;35(11). doi:10.1007/s40520-023-02546-0

187. Molitor RJ, Ko PC, Ally BA. Eye movements in Alzheimer's disease. *Journal of Alzheimer's Disease*. 2015;44(1). doi:10.3233/JAD-141173

Appendices

Appendix 1: DETS Software

```
Option Explicit

Public Sub Main()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim fm As Object

    Set fm = VBA.UserForms.Add("UF_Main")
    fm.Show

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Public Sub InitAppSetts()
    With Application
        .ScreenUpdating = False
        .EnableEvents = False
        .CutCopyMode = False
        .DisplayAlerts = False
        .Calculation = xlCalculationManual
        .WindowState = xlMinimized
    End With
End Sub

Public Sub TermAppSetts()
    With Application
        .ScreenUpdating = True
        .EnableEvents = True
        .CutCopyMode = True
        .DisplayAlerts = True
        .Calculation = xlCalculationAutomatic
    End With
End Sub

Public Sub ClsApp(callerWb As Workbook, callerFmCaption As String, saveWb As Boolean)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, fm As Object

    For Each wb In Application.Workbooks
        If wb.Name <> callerWb.Name Then
            If saveWb Then
                wb.Close SaveChanges:=True
            ElseIf Not saveWb Then
                wb.Close SaveChanges:=False
            End If
        End If
    Next wb

    For Each fm In VBA.UserForms
        If Not fm.Caption = callerFmCaption Then Unload fm
    Next fm
```

```
    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Public Sub ErrorHandler()
    Call InitAppSetts
    Dim FileNumber As Integer
    FileNumber = FreeFile

    Open ThisWorkbook.Path & "\" & "ErrorLog.txt" For Append As #FileNumber
    Print #FileNumber, ">>>>> Error <<<<<"
    Print #FileNumber, "Error occurred at: " & Now
    Print #FileNumber, "Error Number: " & err.Number
    Print #FileNumber, "Error Source: " & err.Source
    Print #FileNumber, "Error Line: " & Erl
    Print #FileNumber, "Error Description: " & err.Description
    Print #FileNumber, "-----------------"
    Close #FileNumber

    MsgBox "Error #" & Str(err.Number) & vbNewLine & _
    " was generated by " & err.Source & "Error Line: " & Erl & vbNewLine & _
    err.Description, vbExclamation, "Error" & vbNewLine & _
    "Error Log created. Please contact administrator: hadikarimi@outlook.com"

    Call ClsApp(ThisWorkbook, "Diagnostic Eye Tracking Study", False)
End Sub

Public Sub ClsForm(fm As UserForm)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim ctl As MSForms.Control, i As Integer

    For Each ctl In fm.Controls
        Select Case TypeName(ctl)
            Case "TextBox"
                ctl.Text = ""
            Case "CheckBox", "OptionButton", "ToggleButton"
                ctl.Value = False
            Case "ComboBox"
                ctl.ListIndex = -1
            Case "ListBox"
                If ctl.MultiSelect = fmMultiSelectMulti Or ctl.MultiSelect = fmMultiSelectExtended
Then
                    For i = 0 To ctl.ListCount - 1
                        ctl.Selected(i) = False
                    Next i
                Else
                    ctl.ListIndex = -1
                End If
        End Select
    Next ctl

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Public Sub Progress()
    Dim fm As Object
    Set fm = VBA.UserForms.Add("UF_Progress")
```

```
    fm.Show
    Call UpdateProgress(0)
End Sub

Public Sub UpdateProgress(addVal As Single)
    Call UF_Progress.Update_Progress(addVal)
    UF_Progress.Show
    DoEvents
End Sub

Option Explicit
    Dim i As Integer, j As Integer
    Dim selectedItemsIndex As Variant

Private Sub UserForm_Initialize()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Call LoadSettings
    ComboBoxEthnic.List = Settings.PtSettings("Ethnicity")
    ComboBoxDx.List = Settings.PtSettings("Diagnosis")
    ListBoxComorbs.List = Settings.PtSettings("Comorbidities")
    BoxNotes.Value = " "
    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub FrameNum_Enter()
    If BoxNum.Value = "0000000000" Then BoxNum.Value = ""
End Sub

Private Sub BoxNum_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
    If Not IsNumeric(Chr(KeyAscii.Value)) And KeyAscii.Value <> 8 Then
        KeyAscii.Value = 0
    End If
End Sub

Private Sub FrameNum_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    BoxNum.Value = Format(BoxNum.Value, "0000000000")
    If BoxNum.Value = "0000000000" Then BoxNum.Value = ""
End Sub

Private Sub FrameDoB_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    If Not IsDate(BoxDoB.Value) Then
        BoxDoB.Value = ""
    Else
        BoxDoB.Value = Format(BoxDoB.Value, "yyyy-mm-dd")
    End If
End Sub

Private Sub FrameDoS_Enter()
    If BoxDoS.Value = Format(Date, "yyyy-mm-dd") Then BoxDoS.Value = ""
End Sub

Private Sub FrameDoS_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    If IsDate(BoxDoS.Value) Then
        BoxDoS.Value = Format(BoxDoS.Value, "yyyy-mm-dd")
    Else
        BoxDoS.Value = Format(Date, "yyyy-mm-dd")
    End If
End Sub
```

```
Private Sub TogButtonMale_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As
Single, ByVal y As Single)
    TogButtonFemale.Value = False
End Sub

Private Sub TogButtonFemale_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As
Single, ByVal y As Single)
    TogButtonMale.Value = False
End Sub

Private Sub BoxMMSE_KeyPress(ByVal KeyAscii As MSForms.ReturnInteger)
    If Not IsNumeric(Chr(KeyAscii.Value)) And KeyAscii.Value <> 8 Then
        KeyAscii.Value = 0
    End If
End Sub

Private Sub FrameMMSE_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    If Not BoxMMSE.Value = "" Then
        If CInt(BoxMMSE.Value) > 30 Or CInt(BoxMMSE.Value) < 1 Then
            BoxMMSE.Value = ""
        End If
    End If
End Sub

Private Sub FrameComorbs_Enter()
    SaveSelectedItems
    FrameComorbs.Height = 86
    ListBoxComorbs.Height = 70
    ListBoxComorbs.IntegralHeight = True
    RestoreSelectedItems
End Sub

Private Sub FrameComorbs_Exit(ByVal Cancel As MSForms.ReturnBoolean)
    SaveSelectedItems
    FrameComorbs.Height = 39
    ListBoxComorbs.IntegralHeight = False
    ListBoxComorbs.Height = 16
    RestoreSelectedItems
End Sub

Private Sub SaveSelectedItems()
    ReDim selectedItemsIndex(1 To ListBoxComorbs.ListCount)
    j = 0
    For i = 0 To ListBoxComorbs.ListCount - 1
        If ListBoxComorbs.Selected(i) Then
            j = j + 1
            selectedItemsIndex(j) = i
        End If
    Next i
End Sub

Private Sub RestoreSelectedItems()
    For i = LBound(selectedItemsIndex) To UBound(selectedItemsIndex)
        If Not IsEmpty(selectedItemsIndex(i)) Then
            ListBoxComorbs.Selected(selectedItemsIndex(i)) = True
        End If
    Next i
End Sub

Private Sub CmdBtnRec_Click()
    'On Error GoTo ErrorHandler
```

```
    Dim ahk As String, ahkFile As String, tobiiManager As String, tobiiLab As String, tobiiProject As
String, tobiPtNum As String

    If IsFormEmpty(Me) Then
        MsgBox ("Please complete information!")
        Exit Sub
    End If

    If Not Settings.PtInfo Is Nothing Then Settings.PtInfo.RemoveAll
    If Not Settings.PtPaths Is Nothing Then Settings.PtPaths.RemoveAll
    Call LoadPtInfo(True)

    ahk = Chr(34) & Settings.AppSettings("AutoHotKey Address") & Chr(34)
    ahkFile = """" & Settings.AppSettings("AutoHotKey Record File Address") & """"
    tobiiManager = Chr(34) & Settings.AppSettings("TobiiPro EyeTracker Manager Address") & Chr(34)
    tobiiLab = Chr(34) & Settings.AppSettings("TobiiPro Lab Address") & Chr(34)
    tobiiProject = Chr(34) & Settings.AppSettings("Tobii DETS project Address") & Chr(34)
    tobiPtNum = Chr(34) & Settings.PtInfo("Number") & Chr(34)

    Shell ahk & " " & ahkFile & " " & tobiiManager & " " & tobiiLab & " " & tobiiProject & " " &
tobiPtNum, vbNormalFocus

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub CmdBtnImpRec_Click()
    'On Error GoTo ErrorHandler
    Dim ahk As String, ahkFile As String, tobiiManager As String, tobiiLab As String, tobiiProject As
String, tobiPtPath As String
    Dim folderObj As Object
    Dim oldFolderPath As String, newFolderPath As String

    If IsFormEmpty(Me) Then
        MsgBox ("Please complete information!")
        Exit Sub
    End If

    If Not Settings.PtInfo Is Nothing Then Settings.PtInfo.RemoveAll
    If Not Settings.PtPaths Is Nothing Then Settings.PtPaths.RemoveAll
    Call LoadPtInfo(True)

    ahk = Chr(34) & Settings.AppSettings("AutoHotKey Address") & Chr(34)
    ahkFile = """" & Settings.AppSettings("AutoHotKey Import File Address") & """"
    tobiiLab = Chr(34) & Settings.AppSettings("TobiiPro Lab Address") & Chr(34)
    tobiiProject = Chr(34) & Settings.AppSettings("Tobii DETS project Address") & Chr(34)
    tobiPtPath = Settings.PtPaths("Patient Data Folder Address")

    If Not IsFolder(Settings.PtPaths("Patient Data Folder Address")) Then
        MkDir Settings.PtPaths("Patient Data Folder Address")
    ElseIf IsFolder(Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS") Then
        Set folderObj = CreateObject("Scripting.FileSystemObject")
        oldFolderPath = Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS"
        newFolderPath = oldFolderPath & "_Old"
        folderObj.MoveFolder oldFolderPath, newFolderPath
    End If

    Shell ahk & " " & ahkFile & " " & tobiiLab & " " & tobiiProject & " " & tobiPtPath, vbNormalFocus

    Exit Sub
ErrorHandler:
```

```
    Call ErrorHandler
End Sub

Private Sub CmdBtnAna_Click()
    'On Error GoTo ErrorHandler
    Dim folderObj As Object, foundFile As Object
    checkPass = True

    If IsFormEmpty(Me) Then
        MsgBox ("Please complete information!")
        Exit Sub
    End If

    If Not Settings.PtInfo Is Nothing Then Settings.PtInfo.RemoveAll
    If Not Settings.PtPaths Is Nothing Then Settings.PtPaths.RemoveAll
    Call LoadPtInfo(True)

    If Not IsFolder(Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS") Then
        MsgBox "No recording data was found!"
        Exit Sub
    Else
        Set folderObj = CreateObject("Scripting.FileSystemObject")
        For Each foundFile In folderObj.GetFolder(Settings.PtPaths("Patient Data Folder Address") &
"Data export - DETS").Files
            If IsEmpty(foundFile) Then
                MsgBox "No recording data was found!"
                Exit Sub
            Else
                Exit For
            End If
        Next foundFile
    End If

    Me.Hide
    Call Progress
    Call Import
    If checkPass Then
        Call Process
        Call DeNoise
        Call Pilot
        Call Analyze
        Call Publish
        Call PlayBeep
    End If
    Call InitAppSetts
    Me.Show

    Call ClsApp(ThisWorkbook, Me.Caption, True)
    Call ClsForm(Me)

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub CmdBtnReAna_Click()
    'On Error GoTo ErrorHandler

    If BoxNum.Value = "" Then
        MsgBox "Please input patient number!"
        Exit Sub
    End If
```

```
    If Not Settings.PtInfo Is Nothing Then Settings.PtInfo.RemoveAll
    If Not Settings.PtPaths Is Nothing Then Settings.PtPaths.RemoveAll
    Call LoadPtInfo(False)

    Me.Hide
    If Not SelectFile("Raw.xlsx") Then
        Me.Show
        Exit Sub
    End If

    Call Progress
    Call Process
    Call DeNoise
    Call Pilot
    Call Analyze
    Call Publish
    Call PlayBeep
    Call InitAppSetts
    Me.Show

    Call ClsForm(Me)
    Call ClsApp(ThisWorkbook, Me.Caption, True)

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub CmdBtnRep_Click()
    'On Error GoTo ErrorHandler

    If BoxNum.Value = "" Then
        MsgBox "Please input patient number!"
        Exit Sub
    End If

    If Not Settings.PtInfo Is Nothing Then Settings.PtInfo.RemoveAll
    If Not Settings.PtPaths Is Nothing Then Settings.PtPaths.RemoveAll
    Call LoadPtInfo(False)

    Me.Hide
    If Not SelectFile("port.pdf") Then
        Me.Show
        Exit Sub
    End If

    Shell "cmd /c start """" """ & Settings.PtPaths("Patient Report File Address") & """"

    Call InitAppSetts
    Me.Show

    Call ClsApp(ThisWorkbook, Me.Caption, True)
    Call ClsForm(Me)

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub CmdBtnDataBank_Click()
    'On Error GoTo ErrorHandler
```

```
    Call ClsApp(ThisWorkbook, Me.Caption, True)
    TermAppSetts
    Shell "cmd /c start """" """ & ThisWorkbook.Path & "\Data_Bank.vbs" & """"
    ThisWorkbook.Saved = True
    Application.Quit

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
    'On Error GoTo ErrorHandler

    Call ClsApp(ThisWorkbook, Me.Caption, True)
    TermAppSetts
    ThisWorkbook.Saved = True
    Application.Quit

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Function SelectFile(fileExt As String) As Boolean
    'On Error GoTo ErrorHandler
    Dim folderObj As Object, foundFile As Object
    Dim listItemsArray As Variant, listArray As Variant, ptPathsArr(1 To 2)  As Variant
    Dim selectedIndex As Integer
    SelectFile = False

    i = 0
    If IsFolder(Settings.PtPaths("Patient Data Folder Address")) Then
        Set folderObj = CreateObject("Scripting.FileSystemObject")
        For Each foundFile In folderObj.GetFolder(Settings.PtPaths("Patient Data Folder
Address")).Files
            If Right$(foundFile.Name, 8) = fileExt Then
                i = i + 1
            End If
        Next
    Else
        MsgBox "Patient Data does not exist!"
        Exit Function
    End If

    If i = 0 Then
        MsgBox "Patient File does not exist!"
        Exit Function
    End If

    ReDim listItemsArray(1 To i)
    ReDim listArray(1 To i)
    j = 0
    For Each foundFile In folderObj.GetFolder(Settings.PtPaths("Patient Data Folder Address")).Files
        If Right$(foundFile.Name, 8) = fileExt Then
            j = j + 1
            listItemsArray(j) = "   " & Left(Right(Settings.PtPaths("Patient Data Folder Address"),
11), 10) & "                    " & Left(foundFile.Name, 10)
            listArray(j) = Left(foundFile.Name, 10)
        End If
    Next
```

```
    selectedIndex = SelectItem(listItemsArray)

    If selectedIndex > 0 Then
        ptPathsArr(1) = Array("Patient Data Folder Address", "Patient Raw Data File Address", "Patient
Processed Data File Address", _
            "Patient Analyzed Data File Address", "Patient Report File Address")
        ptPathsArr(2) = Array(Settings.PtPaths.Item(0), _
            Settings.PtPaths("Patient Data Folder Address") & listArray(selectedIndex) & "_Raw" &
".xlsx", _
            Settings.PtPaths("Patient Data Folder Address") & listArray(selectedIndex) & "_Processed"
& ".xlsx", _
            Settings.PtPaths("Patient Data Folder Address") & listArray(selectedIndex) & "_Analyzed" &
".xlsx", _
            Settings.PtPaths("Patient Data Folder Address") & listArray(selectedIndex) & "_Report" &
".pdf")

        For i = LBound(ptPathsArr(1)) + 1 To UBound(ptPathsArr(1))
            Settings.AddSetting Settings.PtPaths, CStr(ptPathsArr(1)(i)), ptPathsArr(2)(i)
        Next i

        SelectFile = True
    End If

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Sub LoadPtInfo(allFields As Boolean)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim ptInfoArr(1 To 2) As Variant, ptPathsArr(1 To 2) As Variant, selectedListItems As Variant

    ptInfoArr(1) = Array("Number", "Last Name", "First Name", "Birth Date", "Study Date", _
            "Biological Sex", "Ethnicity", "MMSE", "Diagnosis", "Comorbidities", "Notes")
    ptPathsArr(1) = Array("Patient Data Folder Address", "Patient Raw Data File Address", "Patient
Processed Data File Address", _
        "Patient Analyzed Data File Address", "Patient Report File Address")

    If allFields Then
        ReDim selectedListItems(1 To ListBoxComorbs.ListCount)
        j = 0
        For i = 0 To ListBoxComorbs.ListCount - 1
            If ListBoxComorbs.Selected(i) Then
                j = j + 1
                selectedListItems(j) = ListBoxComorbs.List(i)
            End If
        Next i

        ptInfoArr(2) = Array(CStr(BoxNum.Value), CStr(BoxLName.Value), CStr(BoxFName.Value),
CStr(BoxDoB.Value), CStr(BoxDoS.Value), _
            CStr(IIf(TogButtonMale.Value = True, "Male", "Female")), CStr(ComboBoxEthnic.Value),
CStr(BoxMMSE.Value), CStr(ComboBoxDx.Value), _
            selectedListItems, CStr(BoxNotes.Value))

        For i = LBound(ptInfoArr(1)) To UBound(ptInfoArr(1))
            Settings.AddSetting Settings.PtInfo, CStr(ptInfoArr(1)(i)), ptInfoArr(2)(i)
        Next i

        ptPathsArr(2) = Array(Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\", _
```

```
            Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\" & Settings.PtInfo("Study Date") & "_Raw" & ".xlsx", _
            Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\" & Settings.PtInfo("Study Date") & "_Processed" & ".xlsx", _
            Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\" & Settings.PtInfo("Study Date") & "_Analyzed" & ".xlsx", _
            Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\" & Settings.PtInfo("Study Date") & "_Report" & ".pdf")

        For i = LBound(ptPathsArr(1)) To UBound(ptPathsArr(1))
            Settings.AddSetting Settings.PtPaths, CStr(ptPathsArr(1)(i)), ptPathsArr(2)(i)
        Next i
    Else
        ptInfoArr(2) = Array(CStr(BoxNum.Value), "", "", "", "", "", "", "", "", "", "")
        Settings.AddSetting Settings.PtInfo, CStr(ptInfoArr(1)(0)), ptInfoArr(2)(0)

        ptPathsArr(2) = Array(Settings.AppSettings("Patients Recordings Storage Folder Address") &
Settings.PtInfo("Number") & "\", "", "", "", "")
        Settings.AddSetting Settings.PtPaths, CStr(ptPathsArr(1)(0)), ptPathsArr(2)(0)
    End If

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit

Private Sub UserForm_Initialize()
    Call InitAppSetts
    ListBox.Clear
End Sub

Private Sub ButtonOpen_Click()
    If ListBox.ListIndex > -1 Then Me.Hide
End Sub

Private Sub ButtonCancel_Click()
    ListBox.ListIndex = -1
    Me.Hide
End Sub

Public Property Get SelectedItemIndex() As Integer
    SelectedItemIndex = ListBox.ListIndex
End Property

Public Property Let ItemsList(items As Variant)
    ListBox.List = items
End Property

Option Explicit

Public Sub Update_Progress(addVal As Single)
    Dim tempVal As Single

    With Me
        tempVal = .ProgressBar.Value + addVal
        If tempVal > 99 Then tempVal = 99
        .ProgressBar.Value = tempVal
        .TextBoxProgress.Value = "Processing...  " & Round(tempVal, 0) & "%" & " please wait!"
    End With
End Sub
```

```
Option Explicit
    Public Settings As Class_Settings
    Public checkPass As Boolean

Private Sub InitSettings()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Set Settings = New Class_Settings
    Settings.Initialize

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Public Sub LoadSettings()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Call InitSettings

    Dim appSettsArr(1 To 2) As Variant
    appSettsArr(1) = Array("AutoHotKey Address", "AutoHotKey Record File Address", "AutoHotKey Import
File Address", _
        "TobiiPro EyeTracker Manager Address", "TobiiPro Lab Address", "Tobii DETS project Address", _
        "Patients Recordings Storage Folder Address", "Data Bank Address")
    appSettsArr(2) = Array("C:\Program Files\AutoHotkey\v2\AutoHotkey64.exe", "C:\Program
Files\AutoHotkey\DETS\AHK_Record.ahk", "C:\Program Files\AutoHotkey\DETS\AHK_Import.ahk", _

"C:\Users\IALCH_Neurology\AppData\Local\Programs\TobiiProEyeTrackerManager\TobiiProEyeTrackerManager.e
xe", "C:\Program Files\Tobii\Tobii Pro Lab\Bin\TobiiProLab.exe", "C:\Program Files\Tobii\Tobii DETS
Project\tobii.project", _
        "C:\Users\IALCH_Neurology\DETS\Data\", "C:\Users\IALCH_Neurology\DETS\DataBank.xlsx")

    Dim pilotArr(1 To 2) As Variant
    pilotArr(1) = Array( _
        "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down", _
        "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical", _
        "OKN Right", "OKN Left", "OKN Up", "OKN Down", _
        "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal", "AntiSaccades
Vertical")
    pilotArr(2) = Array( _
        Array(0), Array(22), Array(-22), Array(11), Array(-11), _
        Array(22, -22, 6, 3), Array(11, -11, 6, 3), Array(22, -22, 6, 8), Array(11, -11, 6, 4), _
        Array(6), Array(-6), Array(6), Array(-6), _
        Array(9, -9, 9, 3), Array(9, -9, 9, 3), Array(9, -9, 9, 3), Array(9, -9, 9, 3) _
        )

    Dim tablesArr(1 To 2) As Variant
    tablesArr(1) = Array( _
        "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down", _
        "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical", _
        "OKN Right", "OKN Left", "OKN Up", "OKN Down", _
        "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal", "AntiSaccades
Vertical")
    tablesArr(2) = Array( _
        Array("Accuracy", "Variation", "Congruency", "", "", "", "", "", " %", " " & ChrW(176), " " &
ChrW(916) & ChrW(176), "", "", "", "", "", "", " max" & ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "", "", "", "", "", " %", " " & ChrW(176), " " &
ChrW(916) & ChrW(176), "", "", "", "", "", "", " max" & ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "", "", "", "", "", " %", " " & ChrW(176), " " &
ChrW(916) & ChrW(176), "", "", "", "", "", "", " max" & ChrW(176), " %", "", "", "", "", ""), _
```

```
        Array("Accuracy", "Variation", "Congruency", "", "", "", "", "", " %", " " & ChrW(176), " " &
ChrW(916) & ChrW(176), "", "", "", "", "", "", " max" & ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "", "", "", "", "", " %", " " & ChrW(176), " " &
ChrW(916) & ChrW(176), "", "", "", "", "", "", " max" & ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "Latency", "Gain", "Velocity", "", "", " %", " "
& ChrW(176), " " & ChrW(916) & ChrW(176), " ms", "", " " & ChrW(176) & "/s", "", "", "", " max" &
ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "Latency", "Gain", "Velocity", "", "", " %", " "
& ChrW(176), " " & ChrW(916) & ChrW(176), " ms", "", " " & ChrW(176) & "/s", "", "", "", " max" &
ChrW(176), " %", "", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "Phase Shift", "Gain", "", "", "", " %", " " &
ChrW(176), " " & ChrW(916) & ChrW(176), " " & ChrW(176), "", "", "", "", "", " max" & ChrW(176), " %",
"", "", "", "", ""), _
        Array("Accuracy", "Variation", "Congruency", "Phase Shift", "Gain", "", "", "", " %", " " &
ChrW(176), " " & ChrW(916) & ChrW(176), " " & ChrW(176), "", "", "", "", "", " max" & ChrW(176), " %",
"", "", "", "", ""), _
        Array("Gain", "Variation", "", "", "", "", "", "", "", " " & ChrW(176), "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Gain", "Variation", "", "", "", "", "", "", "", " " & ChrW(176), "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Gain", "Variation", "", "", "", "", "", "", "", " " & ChrW(176), "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Gain", "Variation", "", "", "", "", "", "", "", " " & ChrW(176), "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Correct", "Latency", "Error", "", "", "", "", "", "", " ms", "", "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Correct", "Latency", "Error", "", "", "", "", "", "", " ms", "", "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Correct", "Latency", "Error", "", "", "", "", "", "", " ms", "", "", "", "", "", "",
"", "", "", "", "", "", "", "", ""), _
        Array("Correct", "Latency", "Error", "", "", "", "", "", "", " ms", "", "", "", "", "", "",
"", "", "", "", "", "", "", "", "") _
        )

    Dim chartsArr(1 To 2) As Variant
    chartsArr(1) = Array( _
        "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down", _
        "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical", _
        "OKN Right", "OKN Left", "OKN Up", "OKN Down", _
        "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal", "AntiSaccades
Vertical")
    chartsArr(2) = Array( _
        Array("420", "10", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("420", "10", "0", "30", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("420", "10", "-30", "0", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("420", "10", "0", "30", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("420", "10", "-30", "0", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("580", "43", "-30", "30", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("580", "43", "-30", "30", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
```

```
        Array("580", "56", "-30", "30", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("580", "29", "-30", "30", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("350", "20", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("350", "20", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("350", "20", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("350", "20", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("350", "40", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("350", "40", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle"), _
        Array("350", "40", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye
Horizontal Angle", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical Angle"),
_
        Array("350", "40", "-15", "15", "Down", "Up", "Right Eye Vertical Angle", "Left Eye Vertical
Angle", "-15", "15", "Left", "Right", "Right Eye Horizontal Angle", "Left Eye Horizontal Angle") _
        )

    Dim processSettsArr(1 To 2) As Variant
    processSettsArr(1) = Array("Frequency", "Time unit base to Second", _
        "Horizontal pixel count", "Vertical pixel count", _
        "Max Horizontal Angle", "Max Vertical Angle", _
        "Low pass", "High pass", "Threshold", "Tolerance")
    processSettsArr(2) = Array(600, 1000, _
        1920, 1080, _
        22, 11, _
        50, 0.1, 0.1, 1)

    Dim ptSettsArr(1 To 2) As Variant
    ptSettsArr(1) = Array("Ethnicity", "Comorbidities", "Diagnosis")
    ptSettsArr(2) = Array( _
        Array("African", "Asian", "Europian", "Middle Eastern", "Mixed", "Other", "South American"), _
        Array("CVD", "DM", "HTN", "IC", "KD", "LD", "None", _
            "RD", "BMI", "PsD", "RhD", "Smoking"), _
        Array("AIDP", "Atypical Parkinsonism", "Autoimmune Disorder", "Autonomic Disorder", "Brainstem
Disorder", "Cerebellar Disorder", _
            "Channelopathy", "CIDP", "Cognitive Impairment", "COVID-Related Disorder", "Cranial Nerve
Disorder", "Dyssomnia", "Dystonia", _
            "Encephalitis", "Encephalopathy", "Headache", "Hereditary Neuropathy", "HIV-Associated
Disorder", "HTLV-1-Associated Disorder", _
            "Huntington Disease", "Hyperkinesia", "Idiopathic Intracranial Hypertension",
"Intracranial Hypotension", "Ischemic Stroke", _
            "Leukoencephalopathy", "Meningitis", "Metabolic Neuropathy", "Migraine", "Mononeuropathy",
"Mononeuropathy Multiplex", _
            "Motor Neuron Diseases", "Movement Disorder", "MOGAD", "Multiple Sclerosis", "Myelitis",
"Myelopathy", "Neuromuscular Disorder", _
            "Neuropathic Pain", "NMOSD", "Normal", "Normal-Pressure Hydrocephalus", "Parkinson
Disease", "Plexopathy", "Prion Disease", _
            "Sarcoidosis", "Seizure Disorder", "Secondary Malignancy", "Structural Disorder",
"Transient Ischemic Attack", "Tremor", _
            "Vascular Disorders") _
        )
```

```
    Dim i As Long, j As Long
    For i = LBound(appSettsArr(1)) To UBound(appSettsArr(1))
        Settings.AddSetting Settings.AppSettings, CStr(appSettsArr(1)(i)), appSettsArr(2)(i)
    Next i

    For i = LBound(pilotArr(1)) To UBound(pilotArr(1))
        Settings.AddSetting Settings.PilotSettings, CStr(pilotArr(1)(i)), pilotArr(2)(i)
    Next i

    For i = LBound(tablesArr(1)) To UBound(tablesArr(1))
        Settings.AddSetting Settings.TablesSettings, CStr(tablesArr(1)(i)), tablesArr(2)(i)
    Next i

    For i = LBound(chartsArr(1)) To UBound(chartsArr(1))
        Settings.AddSetting Settings.ChartsSettings, CStr(chartsArr(1)(i)), chartsArr(2)(i)
    Next i

    For i = LBound(processSettsArr(1)) To UBound(processSettsArr(1))
        Settings.AddSetting Settings.ProcessSettings, CStr(processSettsArr(1)(i)),
processSettsArr(2)(i)
    Next i

    For i = LBound(ptSettsArr(1)) To UBound(ptSettsArr(1))
        Settings.AddSetting Settings.PtSettings, CStr(ptSettsArr(1)(i)), ptSettsArr(2)(i)
    Next i

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit

Private mAppSetts As Scripting.Dictionary
Private mPilotSetts As Scripting.Dictionary
Private mTablesSetts As Scripting.Dictionary
Private mChartsSetts As Scripting.Dictionary
Private mProcessSetts As Scripting.Dictionary
Private mPtSetts As Scripting.Dictionary
Private mPtInfo As Scripting.Dictionary
Private mPtPaths As Scripting.Dictionary

Public Sub Initialize()
    Set mAppSetts = New Scripting.Dictionary
    Set mPilotSetts = New Scripting.Dictionary
    Set mTablesSetts = New Scripting.Dictionary
    Set mChartsSetts = New Scripting.Dictionary
    Set mProcessSetts = New Scripting.Dictionary
    Set mPtSetts = New Scripting.Dictionary
    Set mPtInfo = New Scripting.Dictionary
    Set mPtPaths = New Scripting.Dictionary
End Sub

Public Property Get AppSettings() As Scripting.Dictionary
    Set AppSettings = mAppSetts
End Property

Public Property Get PilotSettings() As Scripting.Dictionary
    Set PilotSettings = mPilotSetts
End Property

Public Property Get TablesSettings() As Scripting.Dictionary
```

```
    Set TablesSettings = mTablesSetts
End Property

Public Property Get ChartsSettings() As Scripting.Dictionary
    Set ChartsSettings = mChartsSetts
End Property

Public Property Get ProcessSettings() As Scripting.Dictionary
    Set ProcessSettings = mProcessSetts
End Property

Public Property Get PtSettings() As Scripting.Dictionary
    Set PtSettings = mPtSetts
End Property

Public Property Get PtInfo() As Scripting.Dictionary
    Set PtInfo = mPtInfo
End Property

Public Property Get PtPaths() As Scripting.Dictionary
    Set PtPaths = mPtPaths
End Property

Public Sub AddSetting(ByRef dict As Scripting.Dictionary, dickey As String, dicvalue As Variant)
    dict.Add dickey, dicvalue
End Sub

Option Explicit

Public Sub Import()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, ws As Worksheet, srcWB As Workbook, srcWS As Worksheet
    Dim folderObj As Object, srcFile As Object
    Dim wsName As String, oldFolderPath As String, newFolderPath As String
    Dim lastRow As Long, lastCol As Integer, i As Integer, j As Integer
    Dim rawHeaders As Variant, colHeaders As Variant, dataArr As Variant, colData As Variant

    If Not IsFolder(Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS") Then
        MsgBox "No recording data was found!"
        checkPass = False
        Exit Sub
    End If

    If IsFile(Settings.PtPaths("Patient Raw Data File Address")) Then
        MsgBox "Recordings already imported! Please select Re-Analyze."
        checkPass = False
        Exit Sub
    End If

    Set folderObj = CreateObject("Scripting.FileSystemObject")
    For Each srcFile In folderObj.GetFolder(Settings.PtPaths("Patient Data Folder Address") & "Data
export - DETS").Files
        If Not IsFile(srcFile.Path) Then
            MsgBox "No recording data was found!"
            checkPass = False
            Exit Sub
        Else
            Exit For
        End If
    Next srcFile
```

```
    Set wb = Workbooks.Add
    wb.SaveAs Filename:=Settings.PtPaths("Patient Raw Data File Address")
    Set ws = wb.Sheets(1)
    ws.Name = "Report"

    Call FormatIDSheet(ws)

    For Each srcFile In folderObj.GetFolder(Settings.PtPaths("Patient Data Folder Address") & "Data
export - DETS").Files
        Set srcWB = Workbooks.Open(srcFile.Path)
        Set srcWS = srcWB.Sheets(1)
        lastRow = srcWS.UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
        lastCol = srcWS.UsedRange.Columns.Count
        colHeaders = srcWS.Range(srcWS.Cells(1, 1), srcWS.Cells(1, lastCol)).Value2

        For i = 1 To lastCol
            If colHeaders(1, i) = "Timeline name" Then
                wsName = srcWS.Cells(10, i).Value2
                Exit For
            End If
        Next i

        rawHeaders = Array("Recording timestamp", _
            "Gaze point right X", "Gaze point left X", "Gaze point right Y", "Gaze point left Y", _
            "Eye position right Z (DACSmm)", "Eye position left Z (DACSmm)")

        Set ws = wb.Sheets.Add(After:=Sheets(1))
        ws.Name = wsName
        ReDim dataArr(1 To lastRow, 1 To 7)

        For i = LBound(rawHeaders) To UBound(rawHeaders)
            For j = 1 To lastCol
                If rawHeaders(i) = colHeaders(1, j) Then
                    colData = srcWS.Range(srcWS.Cells(1, j), srcWS.Cells(lastRow, j)).Value2
                    dataArr = OverWriteArray(dataArr, colData, i + 1)
                    Exit For
                End If
            Next j
        Next i

        ws.Range(ws.Cells(1, 1), ws.Cells(lastRow, 7)).Value2 = dataArr

        srcWB.Close SaveChanges:=False
        Call UpdateProgress(0.5)
    Next srcFile

    Call Sort_Sheets(wb)

    wb.Save

    Set folderObj = CreateObject("Scripting.FileSystemObject")
    oldFolderPath = Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS"
    newFolderPath = Settings.PtPaths("Patient Data Folder Address") & "Data export - DETS_" &
Settings.PtInfo("Study Date")
    folderObj.MoveFolder oldFolderPath, newFolderPath

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub FormatIDSheet(ws As Worksheet)
```

```
'On Error GoTo ErrorHandler
Call InitAppSetts
Dim i As Integer, tempArr As Variant

With ws
    With .Cells
        .RowHeight = 20
        .ColumnWidth = 16
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlCenter
        .IndentLevel = 0
        .ShrinkToFit = False
        .NumberFormat = "@"
        .WrapText = False
        With .Font
            .Name = "Aptos Display"
            .Size = 12
            .ColorIndex = 1
            .Bold = False
        End With
    End With

    With .Range("$A$1:$I$6")
        .HorizontalAlignment = xlLeft
        .Borders.ColorIndex = 1
        With .Borders(xlEdgeLeft)
            .LineStyle = xlContinuous
            .Weight = xlMedium
        End With
        With .Borders(xlEdgeTop)
            .LineStyle = xlContinuous
            .Weight = xlMedium
        End With
        With .Borders(xlEdgeBottom)
            .LineStyle = xlContinuous
            .Weight = xlMedium
        End With
        With .Borders(xlEdgeRight)
            .LineStyle = xlContinuous
            .Weight = xlMedium
        End With
        .Borders(xlInsideHorizontal).LineStyle = xlNone
        .Borders(xlInsideVertical).LineStyle = xlNone
    End With

    With .Range("$A$1:$I$1")
        .IndentLevel = 1
        .Interior.Color = RGB(65, 50, 125)
        .Font.ColorIndex = 2
    End With

    With .Range("$A$1")
        .Value2 = "DIAGNOSTIC EYE TRACKING STUDY (DETS)"
        .HorizontalAlignment = xlLeft
        With .Font
            .Size = 14
            .Bold = True
        End With
    End With

    With .Range("$I$1")
        .Value2 = "Hadi Karimi"
```

```
            .HorizontalAlignment = xlRight
            With .Font
                .Size = 10
                .Bold = False
            End With
        End With

        With .Range("$A$2:$A$6")
            .IndentLevel = 1
        End With

        .Range("$B$2:$C$6").Font.Bold = True
        .Range("$E$2:$I$6").Font.Bold = True
        .Range("$G$2").Font.Bold = False

        For i = 2 To 6
            .Cells(i, 1).Value2 = Settings.PtInfo.Keys(i - 2) & ":"
            .Cells(i, 2).Value2 = Settings.PtInfo.items(i - 2)
            .Cells(i, 4).Value2 = Settings.PtInfo.Keys(i + 3) & ":"
            If i = 6 Then Exit For
            .Cells(i, 5).Value2 = Settings.PtInfo.items(i + 3)
        Next i

        .Cells(5, 2).Value2 = Format(Settings.PtInfo("Birth Date"), "yyyy-mm-dd")
        .Cells(5, 3).Value2 = Abs(DateDiff("yyyy", Settings.PtInfo("Birth Date"), Date))
        .Cells(6, 2).Value2 = Format(Settings.PtInfo("Study Date"), "yyyy-mm-dd")
        tempArr = Settings.PtInfo("Comorbidities")

        For i = LBound(tempArr) To UBound(tempArr)
            If i > 5 Then Exit For
            .Cells(6, i + 4).Value2 = tempArr(i)
        Next i

        .Cells(2, 7).Value2 = Settings.PtInfo.Keys(10) & ":"
        .Range(Cells(3, 7), Cells(5, 9)).Merge
        .Cells(3, 7).WrapText = True
        .Cells(3, 7).VerticalAlignment = xlTop
        .Cells(3, 7).Value2 = Settings.PtInfo.items(10)
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub Sort_Sheets(wb As Workbook)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim ws As Worksheet, i As Integer, wsOrder As Variant

    wsOrder = Array("Report", "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up",
"Fixation Down", _
        "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical", _
        "OKN Right", "OKN Left", "OKN Up", "OKN Down", _
        "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal", "AntiSaccades
Vertical")

    For i = UBound(wsOrder) To LBound(wsOrder) Step -1
        On Error Resume Next
        Set ws = wb.Sheets(wsOrder(i))
        On Error GoTo 0
        If Not ws Is Nothing Then
```

```
            ws.Move before:=wb.Sheets(1)
            Set ws = Nothing
        End If
    Next i

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit
    Dim calibAngle(1 To 4) As Double

Public Sub Process()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, ws As Worksheet
    Dim freqRowCount As Integer, startRow As Integer, endCol As Integer, endRow As Long
    Dim tolerance As Double
    Dim rawHeaders As Variant, dataArr As Variant

    tolerance = Settings.ProcessSettings("Tolerance")
    freqRowCount = Settings.ProcessSettings("Frequency") * 1.11
    startRow = freqRowCount * 2
    Set wb = Workbooks.Open(Settings.PtPaths("Patient Raw Data File Address"))

    For Each ws In wb.Worksheets
        With ws
            If Not .Name = "Report" Then
                endRow = .UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
                endCol = .Cells(1, .Columns.Count).End(xlToLeft).Column
                rawHeaders = .Range(.Cells(1, 1), .Cells(1, endCol))
                dataArr = .Range(.Cells(startRow, 1), .Cells(endRow, endCol)).Value2

                Call Convert(ws, dataArr, rawHeaders, freqRowCount, startRow, tolerance)
                Call UpdateProgress(0.25)
            End If
        End With
    Next ws

    wb.SaveAs Settings.PtPaths("Patient Processed Data File Address")

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub Convert(ws As Worksheet, ByRef coDataArr As Variant, ByRef coRawHeaders As Variant, ByVal
freqRowCount As Integer, ByVal startRow As Integer, ByVal tolerance As Double)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wsName As String
    Dim lastCol As Integer, lastRow As Long, i As Long, j As Long, k As Long, m As Long
    Dim disAvg(1 To 2) As Double, startTime As Double
    Dim processedHeaders As Variant

    With ws
        coDataArr = FillEmptyData(coDataArr)

        lastRow = UBound(coDataArr, 1)
        lastCol = UBound(coDataArr, 2)
```

```
        k = 0
        For i = lastCol - 1 To lastCol
            k = k + 1
            disAvg(k) = CalcAverage(OneDimArray(coDataArr, i)) * 3.6432
            For j = 1 To lastRow
                If IsEmpty(coDataArr(j, i)) Then
                    coDataArr(j, i) = disAvg(k)
                Else
                    coDataArr(j, i) = coDataArr(j, i) * 3.6432
                End If
            Next j
        Next i

        coDataArr = ConvertToAngle(coDataArr, coRawHeaders)

        processedHeaders = Array("Time", _
            "Right Eye Horizontal Angle", "Left Eye Horizontal Angle", "Right Eye Vertical Angle",
"Left Eye Vertical Angle")

        .UsedRange.ClearContents

        For i = LBound(processedHeaders) To UBound(processedHeaders)
            .Cells(1, i + 1).Value2 = processedHeaders(i)
        Next i

        coDataArr = ResizeArrayShrink(coDataArr, 1, 5, 1, lastRow)

        Call Calibrate(ws, coDataArr, tolerance)
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Function FillEmptyData(ByRef fillDataArr As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, k As Long, m As Long, lastRow As Long, lastCol As Integer
    Dim afterIndex As Long, beforeIndex As Long
    Dim afterValue As Double, beforeValue As Double, Slope As Double

    lastRow = UBound(fillDataArr, 1)
    lastCol = UBound(fillDataArr, 2)

    For i = 1 To lastCol
        For j = 1 To lastRow
            If IsEmpty(fillDataArr(j, i)) Then
                k = j
                Do While IsEmpty(fillDataArr(k, i)) And k < lastRow
                    k = k + 1
                Loop
                If k > lastRow Then
                    afterValue = 0
                    afterIndex = lastRow
                Else
                    afterValue = fillDataArr(k, i)
                    afterIndex = k
                End If

                k = j
                Do While IsEmpty(fillDataArr(k, i)) And k > 1
                    k = k - 1
```

```
                Loop
                If k < 1 Then
                    beforeValue = 0
                    beforeIndex = 1
                Else
                    beforeValue = fillDataArr(k, i)
                    beforeIndex = k
                End If

                If afterIndex - beforeIndex <> 0 Then
                    Slope = (afterValue - beforeValue) / (afterIndex - beforeIndex)
                    For m = beforeIndex + 1 To afterIndex - 1
                        fillDataArr(m, i) = Slope * (m - beforeIndex) + beforeValue
                    Next m
                End If
            End If
        Next j
    Next i

    FillEmptyData = fillDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function ConvertToAngle(ByRef dataArr As Variant, ByRef rawHeaders As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, lastRow As Long, lastCol As Integer, startTime As Double

    lastRow = UBound(dataArr, 1)
    lastCol = UBound(dataArr, 2)

    For i = 1 To UBound(rawHeaders, 2)
        If rawHeaders(1, i) = "Recording timestamp" Then
            startTime = dataArr(1, i)
            For j = 1 To lastRow
                dataArr(j, i) = (dataArr(j, i) - startTime) / Settings.ProcessSettings("Time unit base
to Second")
            Next j
        ElseIf rawHeaders(1, i) = "Gaze point right X" Then
            For j = 1 To lastRow
                dataArr(j, i) = CalcAngle(CDbl((dataArr(j, i) - (Settings.ProcessSettings("Horizontal
pixel count") / 2))), _
                CDbl(dataArr(j, lastCol - 1)))
            Next j
        ElseIf rawHeaders(1, i) = "Gaze point left X" Then
            For j = 1 To lastRow
                dataArr(j, i) = CalcAngle(CDbl((dataArr(j, i) - (Settings.ProcessSettings("Horizontal
pixel count") / 2))), _
                CDbl(dataArr(j, lastCol)))
            Next j
        ElseIf rawHeaders(1, i) = "Gaze point right Y" Then
            For j = 1 To lastRow
                dataArr(j, i) = CalcAngle(CDbl(((Settings.ProcessSettings("Vertical pixel count") / 2)
- dataArr(j, i))), _
                CDbl(dataArr(j, lastCol - 1)))
            Next j
        ElseIf rawHeaders(1, i) = "Gaze point left Y" Then
            For j = 1 To lastRow
                dataArr(j, i) = CalcAngle(CDbl(((Settings.ProcessSettings("Vertical pixel count") / 2)
- dataArr(j, i))), _
```

```
                CDbl(dataArr(j, lastCol)))
            Next j
        End If
    Next i

    ConvertToAngle = dataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcAngle(ByVal angleBase As Double, ByVal angleLeg As Double) As Double
    'On Error GoTo ErrorHandler
    Dim angleTan As Double, angleRadian As Double, angleDegree As Double

    If angleBase = 0 Then
        CalcAngle = 0
    Else
        angleTan = angleBase / angleLeg
        angleRadian = Atn(angleTan)
        angleDegree = angleRadian * (180 / Application.WorksheetFunction.Pi())
        CalcAngle = angleDegree
    End If

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Sub Calibrate(ws As Worksheet, ByRef caDataArr As Variant, ByVal tolerance As Double)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim i As Long, j As Long, lastRow As Long
    Dim avg As Double, diff As Double, w As Double, beta As Double, dNout As Double, tempCalibAngle As
Double, calibAngle(2 To 5) As Double

    With ws
        lastRow = UBound(caDataArr, 1)

        If ws.Name = "Fixation Center" Then
            For i = 2 To 5
                tempCalibAngle = 0 - CalcAverage(OneDimArray(caDataArr, i))
                calibAngle(i) = IIf(tempCalibAngle < tolerance, tempCalibAngle, 0)
            Next i
        End If

        For i = 2 To 5
            For j = 1 To lastRow
                caDataArr(j, i) = caDataArr(j, i) + calibAngle(i)
            Next j
        Next i

        beta = 1
        dNout = 1

        For i = 2 To 5 Step 2
            For j = 1 To lastRow
                avg = (caDataArr(j, i) + caDataArr(j, i + 1)) / 2
                diff = Abs(caDataArr(j, i) - caDataArr(j, i + 1))
                w = 1 / (1 + Exp(-beta * (diff - dNout)))
```

```
                caDataArr(j, i) = caDataArr(j, i) * (1 - w) + avg * w
                caDataArr(j, i + 1) = caDataArr(j, i + 1) * (1 - w) + avg * w
            Next j
        Next i

        .Range(.Cells(2, 1), .Cells(lastRow, 5)).Value2 = caDataArr
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit

Public Sub DeNoise()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, ws As Worksheet
    Dim arrColIndex As Integer, freqRowCount As Integer, lastRow As Long, tolerance As Double,
threshold As Double
    Dim dataArr As Variant, tempArr As Variant

    freqRowCount = Settings.ProcessSettings("Frequency")
    tolerance = Settings.ProcessSettings("Tolerance")
    threshold = Settings.ProcessSettings("Threshold")
    Set wb = Workbooks.Open(Settings.PtPaths("Patient Processed Data File Address"))

    For Each ws In wb.Worksheets
        If Not ws.Name = "Report" Then
            With ws
                lastRow = .UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
                dataArr = .Range(.Cells(2, 2), .Cells(lastRow, 5)).Value2

                For arrColIndex = 1 To 4
                    tempArr = OneDimArray(dataArr, arrColIndex)
                    tempArr = Medianize(tempArr, freqRowCount \ 100)
                    tempArr = DeBlink(tempArr, freqRowCount \ 100, freqRowCount, threshold, tolerance)
                    tempArr = Regressize(tempArr, 385, 5, -4)
                    tempArr = Normalize(tempArr, freqRowCount \ 100)
                    dataArr = OverWriteArray(dataArr, tempArr, arrColIndex)
                    Call UpdateProgress(1)
                Next arrColIndex

                .Range(.Cells(2, 2), .Cells(lastRow, 5)).Value2 = dataArr
            End With
        End If
    Next ws

    wb.Save

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Function Medianize(ByRef medDataArr As Variant, ByVal medWinSize As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, halfWinSize As Integer

    n = UBound(medDataArr)
    halfWinSize = medWinSize \ 2
```

```
    For i = halfWinSize + 1 To n - halfWinSize
        medDataArr(i) = CalcMedian(GetWinArr(medDataArr, i, medWinSize))
    Next i

    Medianize = medDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function DeBlink(ByRef blDataArr As Variant, ByVal blWinSize As Integer, ByVal interval As
Integer, ByVal threshold As Double, ByVal tolerance As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, k As Long, m As Long, n As Long, p As Long, foundBlink As Boolean
    Dim tempSlope As Double, tempAvg As Double, tempVal As Double, slopeArr() As Double, avgArr() As
Double

    ReDim slopeArr(1 To UBound(blDataArr))
    ReDim avgArr(1 To UBound(blDataArr))

    For i = blWinSize * 2 + 1 To UBound(blDataArr) - blWinSize * 2
        tempSlope = CalcSlope(GetWinArr(blDataArr, i - blWinSize, blWinSize * 2))
        tempAvg = CalcAverage(GetWinArr(blDataArr, i - blWinSize, blWinSize * 2))
        If Abs(tempSlope) > threshold And Abs(tempAvg) > tolerance Then
            slopeArr(i) = tempSlope
            avgArr(i) = tempAvg
        Else
            slopeArr(i) = 0
            avgArr(i) = tempAvg
        End If
    Next i

    k = 0
    ReDim blinkPoint(1 To 100)

    For i = blWinSize * 20 + 1 To UBound(blDataArr) - blWinSize * 20
        If slopeArr(i) = 0 Then
            foundBlink = False
            For j = 1 To blWinSize * 20
                If slopeArr(i - j) * slopeArr(i + j) < 0 And Abs(avgArr(i - j) - avgArr(i + j)) <
tolerance Then
                    For m = j To blWinSize * 20
                        If slopeArr(i - m) * slopeArr(i + m) = 0 Then
                            For p = m To blWinSize * 20
                                If slopeArr(i - p) = 0 And slopeArr(i + p) = 0 And Abs(avgArr(i - p) -
avgArr(i + p)) < tolerance Then
                                    foundBlink = True
                                Else
                                    foundBlink = False
                                End If
                            Next p
                            If foundBlink = True Then Exit For
                        End If
                    Next m
                    If foundBlink = True Then Exit For
                End If
            Next j

            If foundBlink = True Then
                If k < 1 Then
                    k = k + 1
```

```
                    blinkPoint(k) = i
                Else
                    If i > blinkPoint(k) + interval Then
                        k = k + 1
                        blinkPoint(k) = i
                    End If
                End If
            End If
        End If
    Next i

    If Not k = 0 Then
        For i = 1 To k
            tempVal = (blDataArr(blinkPoint(i) - blWinSize * 20) + blDataArr(blinkPoint(i) + blWinSize
* 20)) / 2
            For j = blinkPoint(i) - blWinSize * 20 To blinkPoint(i) + blWinSize * 20
                blDataArr(j) = tempVal
            Next j
        Next i
    End If

    DeBlink = blDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function Regressize(ByRef sgDataArr As Variant, ByVal maxWinSize As Integer, _
ByVal minWinSize As Integer, ByVal threshold As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, j As Long, dyWinSize As Integer, edgeAvg As Double, dyWinReg As Variant

    n = UBound(sgDataArr)

    edgeAvg = CalcAverage(GetWinArr(sgDataArr, (maxWinSize \ 4) + 1, maxWinSize \ 2))
    For i = 1 To maxWinSize * 2 \ 3
        sgDataArr(i) = edgeAvg
    Next i

    edgeAvg = CalcAverage(GetWinArr(sgDataArr, n - (maxWinSize \ 4), maxWinSize \ 2))
    For i = n - maxWinSize * 2 \ 3 To n
        sgDataArr(i) = edgeAvg
    Next i

    For i = maxWinSize \ 2 + 1 To n - maxWinSize \ 2
        j = 0
        dyWinSize = maxWinSize
        If dyWinSize Mod 2 = 0 Then dyWinSize = dyWinSize + 1
        Do While dyWinSize >= minWinSize
            j = j + 1
            dyWinReg = CalcRegress(GetWinArr(sgDataArr, i, dyWinSize))
            If dyWinReg(1) > 10 ^ (threshold + j) Then
                dyWinSize = dyWinSize \ 2
                If dyWinSize Mod 2 = 0 Then dyWinSize = dyWinSize + 1
            Else
                sgDataArr(i) = dyWinReg(2)
                Exit Do
            End If
        Loop
    Next i
```

```
    Regressize = sgDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function Normalize(ByRef normDataArr As Variant, ByVal normWinSize As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, halfWinSize As Integer

    n = UBound(normDataArr)
    halfWinSize = normWinSize \ 2
    For i = halfWinSize + 1 To n - halfWinSize
        normDataArr(i) = CalcAverage(GetWinArr(normDataArr, i, normWinSize))
    Next i

    Normalize = normDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Option Explicit

Public Sub Pilot()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, ws As Worksheet, freqRowCount As Integer

    freqRowCount = Settings.ProcessSettings("Frequency") * 1.11
    Set wb = Workbooks.Open(Settings.PtPaths("Patient Processed Data File Address"))

    For Each ws In wb.Worksheets
        If Not ws.Name = "Report" Then
            Call TargetAngle(ws, freqRowCount)
            Call UpdateProgress(0.25)
        End If
    Next ws

    wb.Save

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub TargetAngle(ws As Worksheet, ByVal freqRowCount As Integer)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wsName As String, lastCol As Integer, lastRow As Long, i As Long, j As Long, startRow As Long,
endRow As Long
    Dim dataArr As Variant, tempArr As Variant

    With ws
        wsName = ws.Name
        lastRow = .UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row - 1
        lastCol = .Cells(1, .Columns.Count).End(xlToLeft).Column
        dataArr = .Range(.Cells(2, 2), .Cells(lastRow, 5)).Value2
        ReDim tempArr(1 To lastRow)
        startRow = freqRowCount
```

```
        Select Case wsName
            Case "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down",
_
                "OKN Right", "OKN Left", "OKN Up", "OKN Down", "OKN Horizontal", "OKN Vertical"
                tempArr = FillArray(tempArr, 1, lastRow, CDbl(Settings.PilotSettings(wsName)(0)))

            Case "Saccades Horizontal", "Saccades Vertical"
                tempArr = FillArray(tempArr, 1, startRow, 0)
                For j = 1 To CDbl(Settings.PilotSettings(wsName)(2))
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(0)))
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(1)))
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                Next j
                If Not startRow >= lastRow Then tempArr = FillArray(tempArr, startRow, lastRow, 0)

            Case "Pursuit Horizontal", "Pursuit Vertical"
                tempArr = FillArray(tempArr, 1, startRow, 0)
                endRow = startRow + Settings.PilotSettings(wsName)(2) *
Settings.PilotSettings(wsName)(3) * freqRowCount
                If endRow > lastRow Then endRow = lastRow
                For j = startRow To endRow
                    tempArr(j) = Settings.PilotSettings(wsName)(0) * Sin(2 * WorksheetFunction.Pi() *
(j - startRow) * Settings.PilotSettings(wsName)(2) / (endRow - startRow))
                Next j
                startRow = j
                If Not startRow >= lastRow Then tempArr = FillArray(tempArr, startRow, lastRow, 0)

            Case "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal",
"AntiSaccades Vertical"
                tempArr = FillArray(tempArr, 1, startRow, 0)
                endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                If endRow > lastRow Then endRow = lastRow
                tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(0)))
                startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                For j = 1 To 2
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow, 0)
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(1)))
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                Next j
                For j = 1 To 2
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow, 0)
```

```
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                    endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If endRow > lastRow Then Exit For
                    tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(0)))
                    startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                    If startRow >= lastRow Then Exit For
                Next j
                endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                If endRow > lastRow Then endRow = lastRow
                If Not startRow >= lastRow Then tempArr = FillArray(tempArr, startRow, lastRow, 0)
                startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                endRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                If endRow > lastRow Then endRow = lastRow
                If Not startRow >= lastRow Then tempArr = FillArray(tempArr, startRow, endRow,
CDbl(Settings.PilotSettings(wsName)(1)))
                startRow = startRow + Settings.PilotSettings(wsName)(3) * freqRowCount
                If Not startRow >= lastRow Then tempArr = FillArray(tempArr, startRow, lastRow, 0)
        End Select

        .Cells(1, lastCol + 1).Value2 = "Target Angle"
        .Range(.Cells(2, lastCol + 1), .Cells(lastRow, lastCol + 1)).Value2 = TransposeArray(tempArr)
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit

Public Sub Analyze()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wb As Workbook, ws As Worksheet

    Set wb = Workbooks.Open(Settings.PtPaths("Patient Processed Data File Address"))

    For Each ws In wb.Worksheets
        If Not ws.Name = "Report" Then
            Call Compute(ws)
            Call UpdateProgress(0.5)
        End If
    Next ws

    wb.SaveAs Settings.PtPaths("Patient Analyzed Data File Address")

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub Compute(ws As Worksheet)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim wsName As String, lastCol As Integer, plane As Integer, direct As Integer, freqRowCount As
Integer, rn As Integer
    Dim lastRow As Long, i As Long, j As Long
    Dim tolerance As Double, maxPilot As Double
    Dim dataArr As Variant, headersArr As Variant, tempArr(1 To 8) As Variant, pointsArr As Variant,
acvaArr As Variant, paSaccArr As Variant, oknArr As Variant
```

```
    freqRowCount = Settings.ProcessSettings("Frequency")
    tolerance = Settings.ProcessSettings("Tolerance")

    With ws
        wsName = .Name
        lastRow = .UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
        lastCol = .Cells(1, .Columns.Count).End(xlToLeft).Column
        dataArr = .Range(.Cells(2, 1), .Cells(lastRow, lastCol)).Value2
        maxPilot = Settings.PilotSettings(wsName)(0)

        Select Case wsName
            Case "Fixation Center"
                plane = 0
            Case "Fixation Right", "Fixation Left", "Saccades Horizontal", "Pursuit Horizontal",
"ProSaccades Horizontal", "AntiSaccades Horizontal"
                plane = 1
            Case "Fixation Up", "Fixation Down", "Saccades Vertical", "Pursuit Vertical", "ProSaccades
Vertical", "AntiSaccades Vertical"
                plane = 2
            Case "OKN Right"
                plane = 1
                direct = 1
            Case "OKN Left"
                plane = 1
                direct = 2
            Case "OKN Up"
                plane = 2
                direct = 1
            Case "OKN Down"
                plane = 2
                direct = 2
        End Select

        Select Case wsName
            Case "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down"
                acvaArr = CalcFixAccVar(dataArr, freqRowCount, tolerance, plane)
                tempArr(1) = acvaArr(0)
                tempArr(2) = acvaArr(1)
                tempArr(3) = CalcCongruency(dataArr, tolerance)
                tempArr(4) = Array("", "", "", "", "")
                tempArr(5) = Array("", "", "", "", "")
                tempArr(6) = Array("", "", "", "", "")
                tempArr(7) = Array("", "", "", "", "")
                tempArr(8) = Array("", "", "", "", "")

            Case "Saccades Horizontal", "Saccades Vertical"
                pointsArr = DeflectPoints(dataArr, freqRowCount, maxPilot, plane)
                acvaArr = CalcSacPurAccVar(dataArr, freqRowCount, tolerance, plane)
                tempArr(1) = acvaArr(0)
                tempArr(2) = acvaArr(1)
                tempArr(3) = CalcCongruency(dataArr, tolerance)
                tempArr(4) = CalcLatency(dataArr, pointsArr)
                tempArr(5) = CalcSaccGain(dataArr, pointsArr, freqRowCount, plane, maxPilot)
                tempArr(6) = CalcVelocity(dataArr, pointsArr, freqRowCount, plane)
                tempArr(7) = Array("", "", "", "", "")
                tempArr(8) = Array("", "", "", "", "")

            Case "Pursuit Horizontal", "Pursuit Vertical"
                pointsArr = CrossPoints(dataArr, freqRowCount, plane, 0, freqRowCount)
                acvaArr = CalcSacPurAccVar(dataArr, freqRowCount, tolerance, plane)
                tempArr(1) = acvaArr(0)
```

```
                tempArr(2) = acvaArr(1)
                tempArr(3) = CalcCongruency(dataArr, tolerance)
                tempArr(4) = CalcPhase(dataArr, pointsArr, plane)
                tempArr(5) = CalcPursGain(dataArr, pointsArr, freqRowCount, plane)
                tempArr(6) = Array("", "", "", "", "")
                tempArr(7) = Array("", "", "", "", "")
                tempArr(8) = Array("", "", "", "", "")

            Case "OKN Right", "OKN Left", "OKN Up", "OKN Down"
                pointsArr = BeatPoints(dataArr, freqRowCount, plane)
                oknArr = CalcOKNMetrics(dataArr, pointsArr, plane, direct)
                tempArr(1) = oknArr(0)
                tempArr(2) = oknArr(1)
                tempArr(3) = Array("", "", "", "", "")
                tempArr(4) = Array("", "", "", "", "")
                tempArr(5) = Array("", "", "", "", "")
                tempArr(6) = Array("", "", "", "", "")
                tempArr(7) = Array("", "", "", "", "")
                tempArr(8) = Array("", "", "", "", "")

            Case "ProSaccades Horizontal", "ProSaccades Vertical", "AntiSaccades Horizontal",
"AntiSaccades Vertical"
                pointsArr = StartPoints(dataArr, freqRowCount, plane, 0)
                paSaccArr = CalcProAntiSaccades(dataArr, pointsArr, plane, freqRowCount)
                tempArr(1) = paSaccArr(0)
                tempArr(2) = paSaccArr(1)
                tempArr(3) = paSaccArr(2)
                tempArr(4) = Array("", "", "", "", "")
                tempArr(5) = Array("", "", "", "", "")
                tempArr(6) = Array("", "", "", "", "")
                tempArr(7) = Array("", "", "", "", "")
                tempArr(8) = Array("", "", "", "", "")

        End Select

        headersArr = Settings.TablesSettings(wsName)
        For i = LBound(headersArr) To UBound(headersArr)
            .Cells(1, lastCol + i + 2).Value2 = headersArr(i)
        Next i

        For j = 1 To 8
            For i = 2 To 5
                If IsNumeric(tempArr(j)(i - 1)) Then
                    rn = IIf(j = 5, 2, 1)
                    .Cells(i, lastCol + j + 1).Value2 = Round(tempArr(j)(i - 1), rn)
                Else
                    .Cells(i, lastCol + j + 1).Value2 = tempArr(j)(i - 1)
                End If
            Next i
        Next j
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Function CalcFixAccVar(ByRef avDataArr As Variant, ByVal freqRowCount As Integer, ByVal
tolerance As Double, ByVal plane As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim n As Long, i As Long, j As Long, errRt As Long, errLt As Long
```

```
    Dim diffRtHor As Double, diffLtHor As Double, diffRtVer As Double, diffLtVer As Double, diffRt As
Double, diffLt As Double, _
    maxDiffRt As Double, maxDiffLt As Double, sumDiffRt As Double, sumDiffLt As Double, avgRtHor As
Double, avgRtVer As Double, avgLtHor As Double, avgLtVer As Double
    Dim pPoint As Variant, accuracy(1 To 4) As Variant, variation(1 To 4) As Variant

    n = UBound(avDataArr)

    avgRtHor = CalcAverage(OneDimArray(avDataArr, 2))
    avgLtHor = CalcAverage(OneDimArray(avDataArr, 3))
    avgRtVer = CalcAverage(OneDimArray(avDataArr, 4))
    avgLtVer = CalcAverage(OneDimArray(avDataArr, 5))

    For i = 1 + freqRowCount To n - freqRowCount
        If plane = 0 Then
            diffRtHor = IIf(Abs(avDataArr(i, 2) - 0) < Abs(avDataArr(i, 2) - avgRtHor), avDataArr(i,
2) - 0, avDataArr(i, 2) - avgRtHor)
            diffLtHor = IIf(Abs(avDataArr(i, 3) - 0) < Abs(avDataArr(i, 3) - avgLtHor), avDataArr(i,
3) - 0, avDataArr(i, 3) - avgLtHor)
            diffRtVer = IIf(Abs(avDataArr(i, 4) - 0) < Abs(avDataArr(i, 4) - avgRtVer), avDataArr(i,
4) - 0, avDataArr(i, 4) - avgRtVer)
            diffLtVer = IIf(Abs(avDataArr(i, 5) - 0) < Abs(avDataArr(i, 5) - avgLtVer), avDataArr(i,
5) - 0, avDataArr(i, 5) - avgLtVer)
        ElseIf plane = 1 Then
            diffRtHor = IIf(Abs(avDataArr(i, 2) - avDataArr(i, 6)) < Abs(avDataArr(i, 2) - avgRtHor),
avDataArr(i, 2) - avDataArr(i, 6), avDataArr(i, 2) - avgRtHor)
            diffLtHor = IIf(Abs(avDataArr(i, 3) - avDataArr(i, 6)) < Abs(avDataArr(i, 3) - avgLtHor),
avDataArr(i, 3) - avDataArr(i, 6), avDataArr(i, 3) - avgLtHor)
            diffRtVer = IIf(Abs(avDataArr(i, 4) - 0) < Abs(avDataArr(i, 4) - avgRtVer), avDataArr(i,
4) - 0, avDataArr(i, 4) - avgRtVer)
            diffLtVer = IIf(Abs(avDataArr(i, 5) - 0) < Abs(avDataArr(i, 5) - avgLtVer), avDataArr(i,
5) - 0, avDataArr(i, 5) - avgLtVer)
        ElseIf plane = 2 Then
            diffRtHor = IIf(Abs(avDataArr(i, 2) - 0) < Abs(avDataArr(i, 2) - avgRtHor), avDataArr(i,
2) - 0, avDataArr(i, 2) - avgRtHor)
            diffLtHor = IIf(Abs(avDataArr(i, 3) - 0) < Abs(avDataArr(i, 3) - avgLtHor), avDataArr(i,
3) - 0, avDataArr(i, 3) - avgLtHor)
            diffRtVer = IIf(Abs(avDataArr(i, 4) - avDataArr(i, 6)) < Abs(avDataArr(i, 4) - avgRtVer),
avDataArr(i, 4) - avDataArr(i, 6), avDataArr(i, 4) - avgRtVer)
            diffLtVer = IIf(Abs(avDataArr(i, 5) - avDataArr(i, 6)) < Abs(avDataArr(i, 5) - avgLtVer),
avDataArr(i, 5) - avDataArr(i, 6), avDataArr(i, 5) - avgLtVer)
        End If

        diffRt = Sqr((diffRtHor ^ 2 + diffRtVer ^ 2) / 2)
        diffLt = Sqr((diffLtHor ^ 2 + diffLtVer ^ 2) / 2)

        maxDiffRt = IIf(Abs(diffRt) > Abs(maxDiffRt), IIf(Abs(diffRtHor) > Abs(diffRtVer), diffRtHor,
diffRtVer), maxDiffRt)
        maxDiffLt = IIf(Abs(diffLt) > Abs(maxDiffLt), IIf(Abs(diffLtHor) > Abs(diffLtVer), diffLtHor,
diffLtVer), maxDiffLt)

        sumDiffRt = sumDiffRt + diffRt
        sumDiffLt = sumDiffLt + diffLt

        If diffRt > tolerance Then
            errRt = errRt + 1
        End If
        If diffLt > tolerance Then
            errLt = errLt + 1
        End If
    Next i
```

```
    accuracy(1) = (1 - errRt / n) * 100
    accuracy(2) = ""
    accuracy(3) = (1 - errLt / n) * 100
    accuracy(4) = ""

    variation(1) = sumDiffRt / n
    variation(2) = maxDiffRt
    variation(3) = sumDiffLt / n
    variation(4) = maxDiffLt

    CalcFixAccVar = Array(accuracy, variation)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcSacPurAccVar(ByRef avDataArr As Variant, ByVal freqRowCount As Integer, ByVal
tolerance As Double, ByVal plane As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim n As Long, i As Long, j As Long, errRt As Long, errLt As Long
    Dim diffRtHor As Double, diffLtHor As Double, diffRtVer As Double, diffLtVer As Double, diffRt As
Double, diffLt As Double, _
    maxDiffRt As Double, maxDiffLt As Double, sumDiffRt As Double, sumDiffLt As Double, avgRtHor As
Double, avgRtVer As Double, avgLtHor As Double, avgLtVer As Double
    Dim pPoint As Variant, accuracy(1 To 4) As Variant, variation(1 To 4) As Variant

    n = UBound(avDataArr)

    For i = 1 + freqRowCount To n - freqRowCount
        If plane = 1 Then
            diffRtHor = avDataArr(i, 2) - avDataArr(i, 6)
            diffLtHor = avDataArr(i, 3) - avDataArr(i, 6)

            For j = i - freqRowCount To i + freqRowCount
                diffRtHor = IIf(Abs(avDataArr(i, 2) - avDataArr(j, 6)) < Abs(diffRtHor), avDataArr(i,
2) - avDataArr(j, 6), diffRtHor)
                diffLtHor = IIf(Abs(avDataArr(i, 3) - avDataArr(j, 6)) < Abs(diffLtHor), avDataArr(i,
3) - avDataArr(j, 6), diffLtHor)
            Next j
            diffRtVer = avDataArr(i, 4) - 0
            diffLtVer = avDataArr(i, 5) - 0
        ElseIf plane = 2 Then
            diffRtVer = avDataArr(i, 4) - avDataArr(i, 6)
            diffLtVer = avDataArr(i, 5) - avDataArr(i, 6)
            For j = i - freqRowCount To i + freqRowCount
                diffRtVer = IIf(Abs(avDataArr(i, 4) - avDataArr(j, 6)) < Abs(diffRtVer), avDataArr(i,
4) - avDataArr(j, 6), diffRtVer)
                diffLtVer = IIf(Abs(avDataArr(i, 5) - avDataArr(j, 6)) < Abs(diffLtVer), avDataArr(i,
5) - avDataArr(j, 6), diffLtVer)
            Next j
            diffRtHor = avDataArr(i, 2) - 0
            diffLtHor = avDataArr(i, 3) - 0
        End If

        diffRt = Sqr((diffRtHor ^ 2 + diffRtVer ^ 2) / 2)
        diffLt = Sqr((diffLtHor ^ 2 + diffLtVer ^ 2) / 2)

        maxDiffRt = IIf(Abs(diffRt) > Abs(maxDiffRt), IIf(Abs(diffRtHor) > Abs(diffRtVer), diffRtHor,
diffRtVer), maxDiffRt)
        maxDiffLt = IIf(Abs(diffLt) > Abs(maxDiffLt), IIf(Abs(diffLtHor) > Abs(diffLtVer), diffLtHor,
diffLtVer), maxDiffLt)
```

```
        sumDiffRt = sumDiffRt + diffRt
        sumDiffLt = sumDiffLt + diffLt

        If diffRt > tolerance * 1.5 Then
            errRt = errRt + 1
        End If
        If diffLt > tolerance * 1.5 Then
            errLt = errLt + 1
        End If
    Next i

    accuracy(1) = (1 - errRt / n) * 100
    accuracy(2) = ""
    accuracy(3) = (1 - errLt / n) * 100
    accuracy(4) = ""

    variation(1) = sumDiffRt / n
    variation(2) = maxDiffRt
    variation(3) = sumDiffLt / n
    variation(4) = maxDiffLt

    CalcSacPurAccVar = Array(accuracy, variation)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcCongruency(ByRef cDataArr As Variant, ByVal cTolerance As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim n As Long, i As Long, err As Long, tempVal As Double, sumConVal As Double, congruency(1 To 4)
As Double
    n = UBound(cDataArr)

    err = 0
    sumConVal = 0
    For i = 1 To n
        tempVal = Sqr((cDataArr(i, 2) - cDataArr(i, 3)) ^ 2 + (cDataArr(i, 4) - cDataArr(i, 5)) ^ 2)
        sumConVal = sumConVal + tempVal
        If tempVal > cTolerance Then err = err + 1
    Next i
    congruency(1) = sumConVal / n
    congruency(2) = (1 - err / n) * 100
    congruency(3) = congruency(1)
    congruency(4) = congruency(2)

    CalcCongruency = congruency

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function DeflectPoints(ByRef dpDataArr As Variant, ByVal freqRowCount As Integer, ByVal
maxPilot As Double, ByVal plane As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, n As Long, dPointPi() As Long, dPointRt() As Long, dPointLt() As Long,
addNum As Long
    Dim tempArr As Variant

    ReDim dPointPi(1 To 2, 1 To 20)
```

```
    n = UBound(dpDataArr, 1)
    j = 1

    For i = 2 To n - 1
        If dpDataArr(i, 6) = maxPilot Then
            If dpDataArr(i + 1, 6) = maxPilot And dpDataArr(i - 1, 6) < maxPilot Then
                dPointPi(1, j) = i
            ElseIf dpDataArr(i + 1, 6) < maxPilot And dpDataArr(i - 1, 6) = maxPilot Then
                dPointPi(2, j) = i
                j = j + 1
            End If
        ElseIf dpDataArr(i, 6) = maxPilot * -1 Then
            If dpDataArr(i + 1, 6) = maxPilot * -1 And dpDataArr(i - 1, 6) > maxPilot * -1 Then
                dPointPi(1, j) = i
            ElseIf dpDataArr(i + 1, 6) > maxPilot * -1 And dpDataArr(i - 1, 6) = maxPilot * -1 Then
                dPointPi(2, j) = i
                j = j + 1
            End If
        End If
    Next i

    For i = 1 To UBound(dPointPi, 2)
        If dPointPi(1, i) = 0 And dPointPi(2, i) = 0 Then Exit For
    Next i
    ReDim Preserve dPointPi(1 To 2, 1 To i - 1)
    ReDim dPointRt(1 To 2, 1 To i - 1)
    ReDim dPointLt(1 To 2, 1 To i - 1)

    For i = 1 To UBound(dPointPi, 2)
        For j = 1 To 2
            If dPointPi(j, i) > freqRowCount * 1.5 And dPointPi(j, i) < n - freqRowCount * 1.5 Then
                tempArr = ResizeArrayShrink(dpDataArr, 1, 6, dPointPi(j, i) - freqRowCount * 1.5,
dPointPi(j, i) + freqRowCount * 1.5)
                addNum = dPointPi(j, i) - freqRowCount * 1.5
            ElseIf dPointPi(j, i) <= freqRowCount * 1.5 Then
                tempArr = ResizeArrayShrink(dpDataArr, 1, 6, dPointPi(j, i) - freqRowCount + 1,
dPointPi(j, i) + freqRowCount * 1.5)
                addNum = dPointPi(j, i) - freqRowCount + 1
            ElseIf dPointPi(j, i) >= n - freqRowCount * 1.5 Then
                tempArr = ResizeArrayShrink(dpDataArr, 1, 6, dPointPi(j, i) - freqRowCount * 1.5,
dPointPi(j, i) + freqRowCount - 1)
                addNum = dPointPi(j, i) - freqRowCount * 1.5
            End If

            dPointRt(j, i) = FindEdge(OneDimArray(tempArr, plane * 2), dpDataArr(dPointPi(j, i), 6),
j) + addNum
            dPointLt(j, i) = FindEdge(OneDimArray(tempArr, plane * 2 + 1), dpDataArr(dPointPi(j, i),
6), j) + addNum
        Next j
    Next i

    DeflectPoints = Array(dPointRt, dPointLt, dPointPi)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function FindEdge(ByRef edDataArr As Variant, ByVal maxY As Double, ByVal pointRank As
Integer) As Long
    'On Error GoTo ErrorHandler
    Dim n As Long, i As Long, maxX As Long, edge As Long, dist As Double, tempDist As Double
```

```
    n = UBound(edDataArr)
    dist = 2 * n ^ 2
    maxY = maxY * 1.5
    maxX = 1

    If pointRank = 2 Then maxX = n

    For i = 1 To n
        tempDist = Sqr((maxX - i) ^ 2 + ((maxY - edDataArr(i)) * n / maxY) ^ 2)
        If tempDist < dist Then
            dist = tempDist
            edge = i
        End If
    Next i

    FindEdge = edge

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcLatency(ByRef lDataArr As Variant, ByVal pointsArr As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, nRt As Integer, nLt As Integer, latencyRt As Double, latencyLt As Double
    Dim latency(1 To 4) As Variant, dPointPi As Variant, dPointRt As Variant, dPointLt As Variant

    dPointRt = pointsArr(0)
    dPointLt = pointsArr(1)
    dPointPi = pointsArr(2)

    latencyRt = 0
    latencyLt = 0
    nRt = 0
    nLt = 0

    For i = 1 To UBound(dPointPi, 2)
        If Not dPointRt(2, i) < dPointPi(2, i) Then
            latencyRt = latencyRt + lDataArr(dPointRt(2, i), 1) - lDataArr(dPointPi(2, i), 1)
            nRt = nRt + 1
        End If

        If Not dPointLt(2, i) < dPointPi(2, i) Then
            latencyLt = latencyLt + lDataArr(dPointLt(2, i), 1) - lDataArr(dPointPi(2, i), 1)
            nLt = nLt + 1
        End If
    Next i

    latency(1) = ""
    latency(2) = ""
    latency(3) = ""
    latency(4) = ""
    If Not nRt = 0 Then latency(1) = Round(latencyRt / nRt * 1000, 0)
    If Not nLt = 0 Then latency(3) = Round(latencyLt / nLt * 1000, 0)

    CalcLatency = latency

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function
```

```
Private Function CalcSaccGain(ByRef gDataArr As Variant, ByVal pointsArr As Variant, ByVal
freqRowCount As Integer, ByVal plane As Integer, maxPilot As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, nLRt As Long, nHRt As Long, nLLt As Long, nHLt As Long, pVal As Double
    Dim tempRt As Double, tempLt As Double, gLowRt As Double, gHighRt As Double, gLowLt As Double,
gHighLt As Double
    Dim gain(1 To 4) As Variant, dPointPi As Variant, dPointRt As Variant, dPointLt As Variant

    dPointRt = pointsArr(0)
    dPointLt = pointsArr(1)
    dPointPi = pointsArr(2)

    For i = 1 To UBound(dPointRt, 2)
        For j = dPointRt(1, i) To dPointRt(2, i)
            If dPointRt(2, i) < dPointPi(2, i) + freqRowCount \ 2 Then
                pVal = maxPilot
            Else
                pVal = IIf(j > dPointPi(2, i), 1, maxPilot)
            End If

            tempRt = Abs(gDataArr(j, plane * 2) / pVal)
            If tempRt < 1 Then
                If tempRt < 0.75 Then
                    tempRt = tempRt ^ 2
                Else
                    tempRt = Sqr(tempRt)
                End If
                gLowRt = gLowRt + tempRt
                nLRt = nLRt + 1
            Else
                If tempRt > 1.25 Then
                    tempRt = tempRt ^ 2
                Else
                    tempRt = Sqr(tempRt)
                End If
                gHighRt = gHighRt + tempRt
                nHRt = nHRt + 1
            End If
        Next j
    Next i

    If Not nLRt = 0 Then
        gLowRt = gLowRt / nLRt
    Else
        gLowRt = 1
    End If

    If Not nHRt = 0 Then
        gHighRt = gHighRt / nHRt
    Else
        gHighRt = 1
    End If

    For i = 1 To UBound(dPointLt, 2)
        For j = dPointLt(1, i) To dPointLt(2, i)
            If dPointLt(2, i) < dPointPi(2, i) + freqRowCount \ 2 Then
                pVal = maxPilot
            Else
                pVal = IIf(j > dPointPi(2, i), 1, maxPilot)
            End If
```

```
            tempLt = Abs(gDataArr(j, plane * 2 + 1) / pVal)
            If tempLt < 1 Then
                If tempLt < 0.75 Then
                    tempLt = tempLt ^ 2
                Else
                    tempLt = Sqr(tempLt)
                End If
                gLowLt = gLowLt + tempLt
                nLLt = nLLt + 1
            Else
                If tempLt > 1.25 Then
                    tempLt = tempLt ^ 2
                Else
                    tempLt = Sqr(tempLt)
                End If
                gHighLt = gHighLt + tempLt
                nHLt = nHLt + 1
            End If
        Next j
    Next i

    If Not nLLt = 0 Then
        gLowLt = gLowLt / nLLt
    Else
        gLowLt = 1
    End If

    If Not nHLt = 0 Then
        gHighLt = gHighLt / nHLt
    Else
        gHighLt = 1
    End If

    gain(1) = IIf(1 - gLowRt > gHighRt - 1, gLowRt, gHighRt)
    gain(2) = ""
    gain(3) = IIf(1 - gLowLt > gHighLt - 1, gLowLt, gHighLt)
    gain(4) = ""

    If Round(Abs(1 - gain(1)), 1) = Round(Abs(1 - gain(3)), 1) Then gain(3) = gain(1)

    CalcSaccGain = gain

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcVelocity(ByRef vDataArr As Variant, ByVal pointsArr As Variant, freqRowCount As
Integer, plane As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, nRt As Integer, nLt As Integer, velocityRt As Double, velocityLt As Double
    Dim velocity(1 To 4) As Variant, dPointPi As Variant, dPointRt As Variant, dPointLt As Variant

    dPointRt = pointsArr(0)
    dPointLt = pointsArr(1)
    dPointPi = pointsArr(2)

    For i = 1 To UBound(dPointPi, 2) - 1
        velocityRt = velocityRt + Abs(vDataArr(dPointRt(1, i + 1), plane * 2) - vDataArr(dPointRt(2,
i), plane * 2)) / (vDataArr(dPointRt(1, i + 1), 1) - vDataArr(dPointRt(2, i), 1))
        nRt = nRt + 1
```

```
        velocityLt = velocityLt + Abs(vDataArr(dPointLt(1, i + 1), plane * 2 + 1) -
vDataArr(dPointLt(2, i), plane * 2 + 1)) / (vDataArr(dPointLt(1, i + 1), 1) - vDataArr(dPointLt(2, i),
1))
        nLt = nLt + 1
    Next i

    velocity(1) = ""
    velocity(2) = ""
    velocity(3) = ""
    velocity(4) = ""
    If Not nRt = 0 Then velocity(1) = velocityRt / nRt
    If Not nLt = 0 Then velocity(3) = velocityLt / nLt

    CalcVelocity = velocity

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CrossPoints(ByRef cpDataArr As Variant, ByVal freqRowCount As Integer, ByVal plane As
Integer, ByVal base As Double, ByVal interval As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, n As Long, nPi As Long, nRt As Long, nLt As Long
    Dim cPointPi() As Long, cPointRt() As Long, cPointLt() As Long, tempCPointRt() As Long,
tempCPointLt() As Long
    ReDim cPointPi(1 To 100)
    ReDim cPointRt(1 To 100)
    ReDim cPointLt(1 To 100)
    ReDim tempCPointRt(1 To 100)
    ReDim tempCPointLt(1 To 100)

    n = UBound(cpDataArr, 1)
    nPi = 1
    nRt = 1
    nLt = 1

    For i = 1 + freqRowCount * 2 To n - freqRowCount \ 10 * 2
        If Round(Abs(cpDataArr(i, 6)), 0) = base And Abs(cpDataArr(i, 6)) < Abs(cpDataArr(i - 1, 6))
And Abs(cpDataArr(i, 6)) < Abs(cpDataArr(i + 1, 6)) Then
            For j = 1 To freqRowCount \ 100
                If Abs(cpDataArr(i, 6)) > Abs(cpDataArr(i - j, 6)) Or Abs(cpDataArr(i, 6)) >
Abs(cpDataArr(i + j, 6)) Then Exit For
            Next j
            If j = freqRowCount \ 100 + 1 Then
                cPointPi(nPi) = i
                nPi = nPi + 1
            End If
        End If

        If Round(Abs(cpDataArr(i, plane * 2)), 0) = base And Abs(cpDataArr(i, plane * 2)) <
Abs(cpDataArr(i - 1, plane * 2)) And Abs(cpDataArr(i, plane * 2)) < Abs(cpDataArr(i + 1, plane * 2))
Then
            For j = 1 To freqRowCount \ 100
                If Abs(cpDataArr(i, plane * 2)) > Abs(cpDataArr(i - j, plane * 2)) Or Abs(cpDataArr(i,
plane * 2)) > Abs(cpDataArr(i + j, plane * 2)) Then Exit For
            Next j
            If j = freqRowCount \ 100 + 1 Then
                tempCPointRt(nRt) = i
                nRt = nRt + 1
            End If
        End If
```

```
        If Round(Abs(cpDataArr(i, plane * 2 + 1)), 0) = base And Abs(cpDataArr(i, plane * 2 + 1)) <
Abs(cpDataArr(i - 1, plane * 2 + 1)) And Abs(cpDataArr(i, plane * 2 + 1)) < Abs(cpDataArr(i + 1, plane
* 2 + 1)) Then
            For j = 1 To freqRowCount \ 100
                If Abs(cpDataArr(i, plane * 2 + 1)) > Abs(cpDataArr(i - j, plane * 2 + 1)) Or
Abs(cpDataArr(i, plane * 2 + 1)) > Abs(cpDataArr(i + j, plane * 2 + 1)) Then Exit For
            Next j
            If j = freqRowCount \ 100 + 1 Then
                tempCPointLt(nLt) = i
                nLt = nLt + 1
            End If
        End If
    Next i

    For i = 1 To UBound(tempCPointRt) - 1
        If tempCPointRt(i) = 0 And tempCPointRt(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve tempCPointRt(1 To i)

    For i = 1 To UBound(tempCPointLt)
        If tempCPointLt(i) = 0 And tempCPointLt(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve tempCPointLt(1 To i)

    nRt = 1
    i = 1
    While i <= UBound(tempCPointRt)
        If tempCPointRt(i) + interval < tempCPointRt(i + 1) Or i = UBound(tempCPointRt) Then
            cPointRt(nRt) = tempCPointRt(i)
            nRt = nRt + 1
            i = i + 1
        Else
            For j = i To UBound(tempCPointRt) - 1
                If tempCPointRt(j) + interval < tempCPointRt(j + 1) And tempCPointRt(j) <
cPointPi(nRt) + interval And tempCPointRt(j) > cPointPi(nRt) - interval Then Exit For
            Next j
            If Not j = UBound(tempCPointRt) Then
                cPointRt(nRt) = (tempCPointRt(i) + tempCPointRt(j) + cPointPi(nRt)) \ 3
                nRt = nRt + 1
            Else
                cPointRt(nRt) = 0
                nRt = nRt + 1
            End If
            i = j + 1
        End If
    Wend

    nLt = 1
    i = 1
    While i <= UBound(tempCPointLt)
        If tempCPointLt(i) + interval < tempCPointLt(i + 1) Or i = UBound(tempCPointLt) Then
            cPointLt(nLt) = tempCPointLt(i)
            nLt = nLt + 1
            i = i + 1
        Else
            For j = i To UBound(tempCPointLt) - 1
                If tempCPointLt(j) + interval < tempCPointLt(j + 1) And tempCPointLt(j) <
cPointPi(nLt) + interval And tempCPointLt(j) > cPointPi(nLt) - interval Then Exit For
            Next j
            If Not j = UBound(tempCPointLt) Then
                cPointLt(nLt) = (tempCPointLt(i) + tempCPointLt(j) + cPointPi(nLt)) \ 3
```

```
                nLt = nLt + 1
            Else
                cPointLt(nLt) = 0
                nLt = nLt + 1
            End If
            i = j + 1
        End If
    Wend

    For i = 1 To UBound(cPointPi) - 1
        If cPointPi(i) = 0 And cPointPi(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve cPointPi(1 To i)

    For i = 1 To UBound(cPointRt) - 1
        If cPointRt(i) = 0 And cPointRt(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve cPointRt(1 To i)

    For i = 1 To UBound(cPointLt) - 1
        If cPointLt(i) = 0 And cPointLt(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve cPointLt(1 To i)

    CrossPoints = Array(cPointRt, cPointLt, cPointPi)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcPhase(ByRef phDataArr As Variant, ByVal pointsArr As Variant, ByVal plane As
Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, nRt As Integer, nLt As Integer
    Dim phasRt As Double, phasLt As Double
    Dim phase(1 To 4) As Variant, cPointPi As Variant, cPointRt As Variant, cPointLt As Variant

    cPointRt = pointsArr(0)
    cPointLt = pointsArr(1)
    cPointPi = pointsArr(2)

    For i = 1 To UBound(cPointPi)
        If Not cPointPi(i) = 0 Then
            phasRt = phasRt + (phDataArr(cPointPi(i), plane * 2) - phDataArr(cPointPi(i), 6))
            nRt = nRt + 1
        End If
        If Not cPointPi(i) = 0 Then
            phasLt = phasLt + (phDataArr(cPointPi(i), plane * 2 + 1) - phDataArr(cPointPi(i), 6))
            nLt = nLt + 1
        End If
    Next i

    phase(1) = ""
    phase(2) = ""
    phase(3) = ""
    phase(4) = ""
    If Not nRt = 0 Then phase(1) = phasRt / nRt
    If Not nLt = 0 Then phase(3) = phasLt / nLt

    CalcPhase = phase
```

```
    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcPursGain(ByRef gDataArr As Variant, ByVal pointsArr As Variant, ByVal
freqRowCount As Integer, ByVal plane As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, nLRt As Long, nHRt As Long, nLLt As Long, nHLt As Long, pPoint() As Long
    Dim tempRt As Double, tempLt As Double, gLowRt As Double, gHighRt As Double, gLowLt As Double,
gHighLt As Double
    Dim gain(1 To 4) As Variant, cPointPi As Variant, cPointRt As Variant, cPointLt As Variant

    cPointRt = pointsArr(0)
    cPointLt = pointsArr(1)
    cPointPi = pointsArr(2)
    ReDim pPoint(1 To UBound(cPointPi) - 1)

    For i = 1 To UBound(pPoint)
        pPoint(i) = (cPointPi(i) + cPointPi(i + 1)) \ 2
    Next i

    For i = 1 To UBound(pPoint)
        For j = pPoint(i) - freqRowCount * 2 To pPoint(i) + freqRowCount * 2
            If Abs(gDataArr(j, 6)) > 1 Then
                tempRt = Abs(gDataArr(j, plane * 2) / gDataArr(j, 6))
                If tempRt < 1 Then
                    If tempRt < 0.75 Then
                        tempRt = tempRt ^ 2
                    Else
                        tempRt = Sqr(tempRt)
                    End If
                    gLowRt = gLowRt + tempRt
                    nLRt = nLRt + 1
                Else
                    If tempRt > 1.25 Then
                        tempRt = tempRt ^ 2
                    Else
                        tempRt = Sqr(tempRt)
                    End If
                    gHighRt = gHighRt + tempRt
                    nHRt = nHRt + 1
                End If

                tempLt = Abs(gDataArr(j, plane * 2 + 1) / gDataArr(j, 6))
                If tempLt < 1 Then
                    If tempLt < 0.75 Then
                        tempLt = tempLt ^ 2
                    Else
                        tempLt = Sqr(tempLt)
                    End If
                    gLowLt = gLowLt + tempLt
                    nLLt = nLLt + 1
                Else
                    If tempLt > 1.25 Then
                        tempLt = tempLt ^ 2
                    Else
                        tempLt = Sqr(tempLt)
                    End If
                    gHighLt = gHighLt + tempLt
                    nHLt = nHLt + 1
                End If
```

```
            End If
        Next j
    Next i

    If Not nLRt = 0 Then
        gLowRt = gLowRt / nLRt
    Else
        gLowRt = 1
    End If

    If Not nHRt = 0 Then
        gHighRt = gHighRt / nHRt
    Else
        gHighRt = 1
    End If

    If Not nLLt = 0 Then
        gLowLt = gLowLt / nLLt
    Else
        gLowLt = 1
    End If

    If Not nHLt = 0 Then
        gHighLt = gHighLt / nHLt
    Else
        gHighLt = 1
    End If

    gain(1) = IIf(1 - gLowRt > gHighRt - 1, gLowRt, gHighRt)
    gain(2) = ""
    gain(3) = IIf(1 - gLowLt > gHighLt - 1, gLowLt, gHighLt)
    gain(4) = ""

    If Round(Abs(1 - gain(1)), 1) = Round(Abs(1 - gain(3)), 1) Then gain(3) = gain(1)

    CalcPursGain = gain

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function BeatPoints(ByRef bpDataArr As Variant, ByVal freqRowCount As Integer, ByVal plane As
Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim nMax As Integer, nMin As Integer
    Dim i As Long, j As Long, n As Long, bPointMax() As Long, bPointMin() As Long

    ReDim bPointMax(1 To 20)
    ReDim bPointMin(1 To 20)
    n = UBound(bpDataArr, 1)
    nMax = 1
    nMin = 1

    For i = freqRowCount * 5 To n - freqRowCount * 8
        If bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) > bpDataArr(i - 1, plane * 2) +
bpDataArr(i - 1, plane * 2 + 1) And _
        bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) > bpDataArr(i + 1, plane * 2) +
bpDataArr(i + 1, plane * 2 + 1) And nMax < 21 Then
            For j = 1 To freqRowCount \ 5
                If bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) < bpDataArr(i - j, plane * 2)
+ bpDataArr(i - j, plane * 2 + 1) Or _
```

```
                bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) < bpDataArr(i + j, plane * 2) +
bpDataArr(i + j, plane * 2 + 1) Then Exit For
            Next j
            If j = freqRowCount \ 5 + 1 Then
                bPointMax(nMax) = i
                nMax = nMax + 1
            End If
        End If

        If bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) < bpDataArr(i - 1, plane * 2) +
bpDataArr(i - 1, plane * 2 + 1) And _
        bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) < bpDataArr(i + 1, plane * 2) +
bpDataArr(i + 1, plane * 2 + 1) And nMin < 21 Then
            For j = 1 To freqRowCount \ 5
                If bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) > bpDataArr(i - j, plane * 2)
+ bpDataArr(i - j, plane * 2 + 1) Or _
                bpDataArr(i, plane * 2) + bpDataArr(i, plane * 2 + 1) > bpDataArr(i + j, plane * 2) +
bpDataArr(i + j, plane * 2 + 1) Then Exit For
            Next j
            If j = freqRowCount \ 5 + 1 Then
                bPointMin(nMin) = i
                nMin = nMin + 1
            End If
        End If
    Next i

    For i = 1 To 20
        If bPointMax(i) = 0 Then Exit For
    Next i
    If Not i = 1 Or Not i = 21 Then ReDim Preserve bPointMax(1 To i - 1)

    For i = 1 To 20
        If bPointMin(i) = 0 Then Exit For
    Next i
    If Not i = 1 Or Not i = 21 Then ReDim Preserve bPointMin(1 To i - 1)

    BeatPoints = Array(bPointMax, bPointMin)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcOKNMetrics(ByRef oDataArr As Variant, ByVal pointsArr As Variant, ByVal plane As
Integer, ByVal direct As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, bPMaxVal() As Double, bPMinVal() As Double, bPDiff() As Double
    Dim tempGain As Double, diffSum As Double, diffAvg As Double, tempVar As Double
    Dim gain(1 To 4) As Variant, variation(1 To 4) As Variant

    ReDim bPMaxVal(1 To UBound(pointsArr(0)))
    ReDim bPMinVal(1 To UBound(pointsArr(1)))

    For i = 1 To UBound(bPMaxVal)
        bPMaxVal(i) = (oDataArr(pointsArr(0)(i), plane * 2) + oDataArr(pointsArr(0)(i), plane * 2 +
1)) / 2
    Next i
    For i = 1 To UBound(bPMinVal)
        bPMinVal(i) = (oDataArr(pointsArr(1)(i), plane * 2) + oDataArr(pointsArr(1)(i), plane * 2 +
1)) / 2
    Next i
```

```
    gain(1) = ""
    gain(2) = ""
    gain(3) = ""
    gain(4) = ""
    variation(1) = ""
    variation(2) = ""
    variation(3) = ""
    variation(4) = ""

    n = IIf(UBound(bPMaxVal) < UBound(bPMinVal), UBound(bPMaxVal), UBound(bPMinVal))
    If Not n = 0 Then
        ReDim bPDiff(1 To n)

        For i = 1 To n
            bPDiff(i) = IIf(direct = 1, Abs(bPMaxVal(i) - bPMinVal(i)), Abs(bPMinVal(i) -
bPMaxVal(i)))
            tempGain = tempGain + (bPDiff(i) / 5)
            diffSum = diffSum + bPDiff(i)
        Next i

        diffAvg = diffSum / n

        gain(1) = tempGain / n
        gain(3) = gain(1)


        For i = 1 To n
            tempVar = tempVar + (bPDiff(i) - diffAvg) ^ 2
        Next i

        variation(1) = Sqr(tempVar / n)
        variation(3) = variation(1)
    End If

    CalcOKNMetrics = Array(gain, variation)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function StartPoints(ByRef spDataArr As Variant, ByVal freqRowCount As Integer, ByVal plane As
Integer, ByVal base As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, k As Long, nPi As Long
    Dim sPointPi() As Long, sPointRt() As Long, sPointLt() As Long, avgRtArr() As Double, avgLtArr()
As Double

    ReDim sPointPi(1 To 10)
    nPi = 1

    For i = 1 + freqRowCount \ 2 To UBound(spDataArr, 1) - freqRowCount \ 2
        If Round(Abs(spDataArr(i, 6)), 0) = base And Round(Abs(spDataArr(i + 1, 6)), 0) > base Then
            sPointPi(nPi) = i
            nPi = nPi + 1
        End If
    Next i

    For i = 1 To UBound(sPointPi) - 1
        If sPointPi(i) = 0 And sPointPi(i + 1) = 0 Then Exit For
    Next i
    ReDim Preserve sPointPi(1 To i - 1)
```

```
    ReDim sPointRt(1 To i - 1)
    ReDim sPointLt(1 To i - 1)
    ReDim avgRtArr(1 To UBound(spDataArr, 1))
    ReDim avgLtArr(1 To UBound(spDataArr, 1))

    For i = 1 To UBound(sPointPi)
        For k = sPointPi(i) - freqRowCount To sPointPi(i) + freqRowCount * 2
            avgRtArr(k) = CalcAverage(GetWinArr(OneDimArray(spDataArr, plane * 2), k, freqRowCount \
100))
            avgLtArr(k) = CalcAverage(GetWinArr(OneDimArray(spDataArr, plane * 2 + 1), k, freqRowCount
\ 100))
        Next k

        For k = sPointPi(i) - freqRowCount To sPointPi(i) + freqRowCount * 2
            If Abs(Round(avgRtArr(k), 0)) < Abs(Round(avgRtArr(k + 1), 0)) Then
                For j = k + 1 To k + freqRowCount \ 100
                    If Not Abs(Round(avgRtArr(j), 1)) < Abs(Round(avgRtArr(j + 1), 1)) Then Exit For
                Next j
                If j > k + freqRowCount \ 100 Then
                    sPointRt(i) = k
                    Exit For
                End If
            End If
        Next k

        For k = sPointPi(i) - freqRowCount To sPointPi(i) + freqRowCount * 2
            If Abs(Round(avgLtArr(k), 0)) < Abs(Round(avgLtArr(k + 1), 0)) Then
                For j = k + 1 To k + freqRowCount \ 100
                    If Not Abs(Round(avgLtArr(j), 1)) < Abs(Round(avgLtArr(j + 1), 1)) Then Exit For
                Next j
                If j > k + freqRowCount \ 100 Then
                    sPointLt(i) = k
                    Exit For
                End If
            End If
        Next k
    Next i

    StartPoints = Array(sPointRt, sPointLt, sPointPi)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Private Function CalcProAntiSaccades(ByRef paDataArr As Variant, ByVal pointsArr As Variant, ByVal
plane As Integer, ByVal freqRowCount As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, nRt As Integer, nLt As Integer, nCRt As Integer, nCLt As Integer,
latencyCRt As Double, latencyCLt As Double
    Dim correct(1 To 4) As Variant, cLatency(1 To 4) As Variant, err(1 To 4) As Variant, sPointPi As
Variant, sPointRt As Variant, sPointLt As Variant

    sPointRt = pointsArr(0)
    sPointLt = pointsArr(1)
    sPointPi = pointsArr(2)

    latencyCRt = 0
    latencyCLt = 0
    nCRt = 0
    nCLt = 0
    nRt = 0
```

```
    nLt = 0

    For i = 1 To UBound(sPointPi)
        If Not sPointRt(i) < sPointPi(i) And Not sPointPi(i) = 0 Then
            nRt = nRt + 1
            latencyCRt = latencyCRt + paDataArr(sPointRt(i), 1) - paDataArr(sPointPi(i), 1)

            If CalcSlope(GetWinArr(OneDimArray(paDataArr, plane * 2), sPointRt(i) + freqRowCount \ 20,
freqRowCount \ 10)) * _
            CalcSlope(GetWinArr(OneDimArray(paDataArr, 6), sPointPi(i) + freqRowCount \ 20,
freqRowCount \ 10)) > 0 And _
            Abs(Round(CalcAverage(GetWinArr(OneDimArray(paDataArr, plane * 2), sPointRt(i) +
freqRowCount * 3 \ 2, freqRowCount * 3)), 0)) > _
            Abs(Round(CalcAverage(GetWinArr(OneDimArray(paDataArr, 6), sPointPi(i) + freqRowCount * 3
\ 2, freqRowCount * 3)), 0)) * 2 / 3 Then nCRt = nCRt + 1
        End If

        If Not sPointLt(i) < sPointPi(i) And Not sPointPi(i) = 0 Then
            nLt = nLt + 1
            latencyCLt = latencyCLt + paDataArr(sPointLt(i), 1) - paDataArr(sPointPi(i), 1)

            If CalcSlope(GetWinArr(OneDimArray(paDataArr, plane * 2 + 1), sPointLt(i) + freqRowCount \
20, freqRowCount \ 10)) * _
            CalcSlope(GetWinArr(OneDimArray(paDataArr, 6), sPointPi(i) + freqRowCount \ 20,
freqRowCount \ 10)) > 0 And _
            Abs(Round(CalcAverage(GetWinArr(OneDimArray(paDataArr, plane * 2 + 1), sPointLt(i) +
freqRowCount * 3 \ 2, freqRowCount * 3)), 0)) > _
            Abs(Round(CalcAverage(GetWinArr(OneDimArray(paDataArr, 6), sPointPi(i) + freqRowCount * 3
\ 2, freqRowCount * 3)), 0)) * 2 / 3 Then nCLt = nCLt + 1
        End If
    Next i

    correct(1) = nCRt
    correct(2) = ""
    correct(3) = nCLt
    correct(4) = ""

    cLatency(1) = ""
    If Not nRt = 0 Then cLatency(1) = Round(latencyCRt / nRt * 1000, 0)
    cLatency(2) = ""
    cLatency(3) = ""
    If Not nLt = 0 Then cLatency(3) = Round(latencyCLt / nLt * 1000, 0)
    cLatency(4) = ""

    err(1) = 6 - nCRt
    err(2) = ""
    err(3) = 6 - nCLt
    err(4) = ""

    CalcProAntiSaccades = Array(correct, cLatency, err)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Option Explicit

Public Sub Publish()
    'On Error GoTo ErrorHandler
    Call InitAppSetts
```

```
    Dim wb As Workbook, reportWS As Worksheet, ws As Worksheet, breakRow As Integer, prevWSName As
String

    Set wb = Workbooks.Open(Settings.PtPaths("Patient Analyzed Data File Address"))
    Set reportWS = wb.Sheets("Report")

    breakRow = 7

    For Each ws In wb.Worksheets
        If Not ws.Name = "Report" Then
            Call Charts(reportWS, ws, breakRow)
            Call Tables(reportWS, ws, breakRow)
            Call UpdateProgress(0.25)

            If ws.Name = "OKN Right" And prevWSName = "Pursuit Vertical" Then
                breakRow = breakRow
            ElseIf ws.Name = "OKN Up" And prevWSName = "OKN Left" Then
                breakRow = breakRow
            ElseIf ws.Name = "ProSaccades Horizontal" And prevWSName = "OKN Down" Then
                breakRow = breakRow
            ElseIf ws.Name = "AntiSaccades Horizontal" And prevWSName = "ProSaccades Vertical" Then
                breakRow = breakRow
            Else
                breakRow = breakRow + 55
                reportWS.HPageBreaks.Add reportWS.Range("A" & breakRow)
            End If

            prevWSName = ws.Name
        End If
    Next ws

    wb.Save

    Call DataBank(wb)
    Call UpdateProgress(0.25)
    Call Report(reportWS)

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub Report(ByVal ws As Worksheet)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim printRng As String, titleRng As String, wsName As String
    Dim lastRow As Long, lastCol As Integer

    With ws
        lastRow = .UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
        lastCol = .Cells(1, .Columns.Count).End(xlToLeft).Column
        printRng = .Range(.Cells(1, 1), .Cells(lastRow, lastCol)).Address
        titleRng = .Range(.Cells(1, 1), .Cells(6, lastCol)).Address
    End With

    Application.PrintCommunication = True
    With ws.PageSetup
        .PrintArea = printRng
        .PrintTitleRows = titleRng
    End With

    Application.PrintCommunication = False
```

```
    With ws.PageSetup
        .PrintArea = printRng
        .PrintTitleRows = titleRng
        .PaperSize = xlPaperA4
        .Orientation = xlPortrait
        .Zoom = 100
        .FitToPagesWide = 1
        .FitToPagesTall = False
        .LeftMargin = Application.InchesToPoints(0.6)
        .RightMargin = Application.InchesToPoints(0.6)
        .TopMargin = Application.InchesToPoints(0.7)
        .BottomMargin = Application.InchesToPoints(0.7)
        .HeaderMargin = Application.InchesToPoints(0)
        .FooterMargin = Application.InchesToPoints(0)
        .CenterHorizontally = True
        .CenterVertically = False
    End With

    Application.PrintCommunication = True
    ws.ExportAsFixedFormat Type:=xlTypePDF, Filename:=Settings.PtPaths("Patient Report File Address"),
Quality:=xlQualityMinimum, _
        IncludeDocProperties:=False, IgnorePrintAreas:=False, OpenAfterPublish:=True

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub DataBank(ByRef wb As Workbook)
    ''On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim dbWB As Workbook, dbWS As Worksheet, ws As Worksheet
    Dim lastRow As Integer, rowNum As Integer, i As Integer, j As Integer, k As Integer, m As Integer,
sCount As Integer

    Set dbWB = Workbooks.Open(Settings.AppSettings("Data Bank Address"))
    Set dbWS = dbWB.Sheets("DataBank")
    Set ws = wb.Sheets("Report")

    lastRow =
dbWS.ListObjects("DataBank").ListColumns("Number").Range.Cells(dbWS.ListObjects("DataBank").ListColumn
s("Number").Range.Cells.Count).End(xlUp).Row

    For i = 1 To lastRow
        If Settings.PtInfo("Number") = dbWS.Cells(i, 2) Then
            rowNum = i
        Else
            rowNum = IIf(lastRow = 2, 3, lastRow + 4)
        End If
    Next i

    For i = 2 To 11
        If i < 6 Then
            dbWS.Cells(rowNum, i).Value2 = ws.Cells(i, 2).Value2
        ElseIf i = 6 Then
            dbWS.Cells(rowNum, i).Value2 = ws.Cells(i - 1, 3).Value2
        ElseIf i = 7 Then
            dbWS.Cells(rowNum, i).Value2 = ws.Cells(i - 1, 2).Value2
        Else
            dbWS.Cells(rowNum, i).Value2 = ws.Cells(i - 6, 5).Value2
        End If
    Next i
```

```
    For i = 1 To 4
        dbWS.Cells(rowNum + i - 1, 12).Value2 = ws.Cells(6, 5 + i - 1).Value2
    Next i

    dbWS.Cells(rowNum, 13).Value2 = ws.Cells(3, 7).Value2

    For Each ws In wb.Worksheets
        sCount = sCount + 1
    Next ws

    m = 15

    For i = 2 To sCount
        Set ws = wb.Sheets(i)
        Select Case Left(ws.Name, 3)
            Case "Fix"
                For k = m To m + 2
                    For j = 1 To 4
                        dbWS.Cells(rowNum + j - 1, k).Value2 = ws.Cells(j + 1, k - m + 8).Value2
                    Next j
                Next k
                m = m + 3
            Case "Sac"
                For k = m To m + 5
                    For j = 1 To 4
                        dbWS.Cells(rowNum + j - 1, k).Value2 = ws.Cells(j + 1, k - m + 8).Value2
                    Next j
                Next k
                m = m + 6
            Case "Pur"
                For k = m To m + 4
                    For j = 1 To 4
                        dbWS.Cells(rowNum + j - 1, k).Value2 = ws.Cells(j + 1, k - m + 8).Value2
                    Next j
                Next k
                m = m + 5
            Case "OKN"
                For k = m To m + 1
                    For j = 1 To 4
                        dbWS.Cells(rowNum + j - 1, k).Value2 = ws.Cells(j + 1, k - m + 8).Value2
                    Next j
                Next k
                m = m + 2
            Case "Pro", "Ant"
                For k = m To m + 2
                    For j = 1 To 4
                        dbWS.Cells(rowNum + j - 1, k).Value2 = ws.Cells(j + 1, k - m + 8).Value2
                    Next j
                Next k
                m = m + 3
        End Select
    Next i

    dbWB.Close SaveChanges:=True

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit
```

```
Public Sub Tables(ByRef reportWS As Worksheet, ByRef ws As Worksheet, ByVal breakRow)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim tblName As String, tblPos As Integer, tblRows As Integer

    tblRows = 5
    tblName = ws.Name

    Select Case tblName
        Case "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down", _
            "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical", _
            "OKN Left", "OKN Down", "ProSaccades Vertical", "AntiSaccades Vertical"
                tblPos = breakRow + 49
        Case "OKN Right", "OKN Up", "ProSaccades Horizontal", "AntiSaccades Horizontal"
            tblPos = breakRow + 21
    End Select

    Call FormatTables(reportWS, ws, tblName, tblPos, tblRows)
    Call FillTables(reportWS, ws, "RepTbl_" & tblName)

    Exit Sub
ErrorHandler:
    ErrorHandler
End Sub

Private Sub FormatTables(ByRef reportWS As Worksheet, ByRef ws As Worksheet, ByVal tblName As String,
ByVal tblPos As Integer, ByVal tblRows As Integer)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim tbl As ListObject, colNum As Integer, rowNum As Integer, i As Integer

    With reportWS.Range(reportWS.Cells(tblPos, 1), reportWS.Cells(tblPos, 9))
        .Merge
        .Value2 = tblName
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlCenter
        .WrapText = False
        .ShrinkToFit = False
        .ReadingOrder = xlContext
        .Borders.Weight = xlThin
        .Borders(xlInsideVertical).LineStyle = xlNone
        .Borders(xlInsideHorizontal).LineStyle = xlNone
        .Font.Name = "Aptos Display"
        .Font.Size = 12
        .Font.Bold = True
        .Interior.Color = RGB(180, 180, 205)
    End With

    Set tbl = reportWS.ListObjects.Add(xlSrcRange, reportWS.Range(reportWS.Cells(tblPos + 1, 1),
reportWS.Cells(tblPos + tblRows, 9)), , xlYes)
    With tbl
        .Name = "RepTbl_" & tblName
        .ShowAutoFilterDropDown = False
        .ShowTableStyleRowStripes = False
        .ListColumns(1).DataBodyRange(tblRows \ 4) = "Right Eye"
        .ListColumns(1).DataBodyRange(tblRows \ 4 + tblRows \ 2) = "Left Eye"

        .HeaderRowRange(1).Font.Color = RGB(205, 205, 230)

        For colNum = 2 To 9
            .ListColumns(colNum).Name = Settings.TablesSettings(ws.Name)(colNum - 2)
```

```
            If Settings.TablesSettings(ws.Name)(colNum - 2) = "" Then
.HeaderRowRange(colNum).Font.Color = RGB(205, 205, 230)
        Next colNum

        With .Range
            .HorizontalAlignment = xlCenter
            .VerticalAlignment = xlCenter
            .WrapText = False
            .ShrinkToFit = False
            .ReadingOrder = xlContext
            .Borders.LineStyle = xlContinuous
            .Font.Name = "Aptos Display"
            .Font.Size = 11
            .Font.Bold = False
        End With

        With .HeaderRowRange
            .Font.Name = "Aptos Display"
            .Font.Size = 11
            .Font.Bold = True
            .Interior.Color = RGB(205, 205, 230)
        End With

        With .ListColumns(1).DataBodyRange
            .Font.Size = 11
            .Font.Bold = True
        End With

        For rowNum = 1 To tblRows \ 2
            .ListRows(rowNum).Range.Interior.Color = RGB(255, 250, 250)
            .ListRows(rowNum + tblRows \ 2).Range.Interior.Color = RGB(250, 250, 255)
        Next rowNum

        For rowNum = 1 To tblRows \ 2 - 1
            .ListRows(rowNum).Range.Borders(xlEdgeBottom).Color = RGB(255, 250, 250)
            .ListRows(rowNum + tblRows \ 2).Range.Borders(xlEdgeBottom).Color = RGB(250, 250, 255)
        Next rowNum
    End With

    Exit Sub
ErrorHandler:
    ErrorHandler
End Sub

Private Sub FillTables(reportWS As Worksheet, ws As Worksheet, ByVal tblName As String)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim tbl As ListObject, wsName As String, i As Long, j As Long
    Dim metricsArr As Variant, symbolArr(1 To 3) As Variant

    metricsArr = ws.Range(ws.Cells(2, 8), ws.Cells(9, 16)).Value2

    Set tbl = reportWS.ListObjects(tblName)
    With tbl
        For i = 2 To 9
            For j = 1 To 3 Step 2
                .ListColumns(i).DataBodyRange(j).Value2 = metricsArr(j, i - 1) &
Settings.TablesSettings(ws.Name)(i + 6)
            Next j
            For j = 2 To 4 Step 2
                .ListColumns(i).DataBodyRange(j).Value2 = metricsArr(j, i - 1) &
Settings.TablesSettings(ws.Name)(i + 14)
```

```
            Next j
        Next i
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Option Explicit

Public Sub Charts(ByRef reportWS As Worksheet, ByRef ws As Worksheet, ByVal breakRow As Integer)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim chrt As Chart, dashLine As Boolean
    Dim chrtTitle As String, yAxTitle1 As String, yAxTitle2 As String
    Dim lastRow As Long
    Dim chrtPos(1 To 2) As Integer, chrtHeight(1 To 2) As Integer, arrColIndex(1 To 2) As Integer, _
    xAxMax As Integer, yAxMin As Integer, yAxMax As Integer, i As Integer, j As Integer, k As Integer
    Dim chrtDataArr As Variant

    lastRow = ws.UsedRange.Cells.SpecialCells(xlCellTypeLastCell).Row
    chrtDataArr = ws.Range(ws.Cells(2, 1), ws.Cells(lastRow, 6)).Value2

    Select Case ws.Name
        Case "Fixation Center", "Fixation Right", "Fixation Left", "Fixation Up", "Fixation Down"
            j = 2
            chrtPos(1) = breakRow + 2
            chrtPos(2) = breakRow + 25
            chrtHeight(1) = Settings.ChartsSettings(ws.Name)(0)
            chrtHeight(2) = Settings.ChartsSettings(ws.Name)(0)
            arrColIndex(1) = 6
            arrColIndex(2) = 0
        Case "Saccades Horizontal", "Saccades Vertical", "Pursuit Horizontal", "Pursuit Vertical"
            j = 2
            chrtPos(1) = breakRow + 2
            chrtPos(2) = breakRow + 33
            chrtHeight(1) = Settings.ChartsSettings(ws.Name)(0)
            chrtHeight(2) = Settings.ChartsSettings(ws.Name)(0) \ 2
            arrColIndex(1) = 6
            arrColIndex(2) = 0
        Case "OKN Right", "OKN Up", "ProSaccades Horizontal", "AntiSaccades Horizontal"
            j = 1
            chrtPos(1) = breakRow + 2
            chrtHeight(1) = Settings.ChartsSettings(ws.Name)(0)
            arrColIndex(1) = 6
        Case "OKN Left", "OKN Down", "ProSaccades Vertical", "AntiSaccades Vertical"
            j = 1
            chrtPos(1) = breakRow + 29
            chrtHeight(1) = Settings.ChartsSettings(ws.Name)(0)
            arrColIndex(1) = 6
    End Select

    For i = 1 To j
        chrtTitle = ws.Name & IIf(Settings.ChartsSettings(ws.Name)(i * 6) = "Right Eye Horizontal
Angle", " - Horizontal Axis", " - Vertical Axis")
        k = IIf(Settings.ChartsSettings(ws.Name)(i * 6) = "Right Eye Horizontal Angle", 1, 2)
        xAxMax = Settings.ChartsSettings(ws.Name)(1)
        yAxMin = Settings.ChartsSettings(ws.Name)(2 * i * i)
        yAxMax = Settings.ChartsSettings(ws.Name)(2 * i * i + 1)
        yAxTitle1 = Settings.ChartsSettings(ws.Name)((i + 3) * i)
        yAxTitle2 = Settings.ChartsSettings(ws.Name)((i + 3) * i + 1)
```

```
        Set chrt = reportWS.Shapes.AddChart2(240, xlXYScatterSmoothNoMarkers, _
            1, reportWS.Range("$A$" & chrtPos(i)).Top, reportWS.Range("$J$" & chrtPos(i)).Left,
chrtHeight(i), True).Chart

        Call FormatCharts(chrt, chrtTitle, yAxTitle1, yAxTitle2, yAxMin, yAxMax, xAxMax)

        Call ChartSeries(chrt, "Target Pilot", RGB(0, 0, 0), True, chrtDataArr, arrColIndex(i))

        Call ChartSeries(chrt, Settings.ChartsSettings(ws.Name)(i * 6 + 1), RGB(0, 0, 255), False,
chrtDataArr, k * 2 + 1)

        Call ChartSeries(chrt, Settings.ChartsSettings(ws.Name)(i * 6), RGB(255, 0, 0), False,
chrtDataArr, k * 2)
    Next i

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub FormatCharts(ByRef chrt As Chart, ByVal chrtTitle As String, ByVal yAxTitle1 As String,
ByVal yAxTitle2 As String, _
    ByVal yAxMin As Integer, ByVal yAxMax As Integer, ByVal xAxMax As Integer)
    'On Error GoTo ErrorHandler
    Call InitAppSetts

    With chrt
        .Parent.RoundedCorners = True
        .SetElement (msoElementChartTitleAboveChart)
        .SetElement (msoElementPrimaryCategoryAxisShow)
        .SetElement (msoElementPrimaryCategoryAxisTitleAdjacentToAxis)
        .SetElement (msoElementPrimaryValueAxisShow)
        .SetElement (msoElementPrimaryValueAxisTitleAdjacentToAxis)
        .SetElement (msoElementDataLabelNone)
        .SetElement (msoElementLegendTop)
        With .ChartArea
            .ClearContents
            With .Border
                .Weight = 1
                .ColorIndex = 1
            End With
            With .Font
                .Name = "Aptos Display"
                .Size = 11
                .ColorIndex = 1
            End With
        End With
        With .ChartTitle
            .Text = chrtTitle
            With .Font
                .Size = 12
                .Bold = True
                .ColorIndex = 1
            End With
        End With
        With .Axes(xlCategory, xlPrimary)
            .HasMinorGridlines = True
            .HasMajorGridlines = True
            .MinimumScale = 0
            .MaximumScale = xAxMax
            .MajorUnit = IIf(xAxMax <= 20, 1, 2)
```

```
            .MinorUnit = IIf(xAxMax <= 20, 0.1, 0.2)
            .TickLabelPosition = xlLow
            .TickLabels.Font.ColorIndex = 1
            .HasTitle = True
            .AxisTitle.Characters.Text = "Time (s)"
        End With
        With .Axes(xlValue, xlPrimary)
            .HasMajorGridlines = True
            .HasMinorGridlines = True
            .MinimumScale = yAxMin
            .MaximumScale = yAxMax
            .MinorUnit = 1
            .MajorUnit = 5
            .TickLabels.Font.ColorIndex = 1
            .HasTitle = True
            .AxisTitle.Characters.Text = yAxTitle1 & "  <--  " & "Angle (" & ChrW(176) & ")" & "  -->
" & yAxTitle2
        End With
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Private Sub ChartSeries(ByRef chrt As Chart, ByVal serName As String, ByVal serColor As Long, ByVal
dashLine As Boolean, _
    ByRef chrtDataArr As Variant, ByVal arrColIndex As Integer)
    'On Error GoTo ErrorHandler
    Call InitAppSetts
    Dim ser As Series

    Set ser = chrt.SeriesCollection.NewSeries
    With ser
        .Name = serName
        .AxisGroup = xlPrimary
        .Smooth = True
        .MarkerStyle = xlMarkerStyleNone
        With .Format.Line
            .Visible = msoTrue
            .ForeColor.RGB = serColor
            .Weight = 1.5
            .DashStyle = IIf(dashLine, msoLineDashDot, msoLineSolid)
        End With
        .XValues = OneColTwoDimArray(chrtDataArr, 1)
        If arrColIndex = 0 Then
            Dim nullArr As Variant, n As Long, i As Long
            n = UBound(chrtDataArr, 1)
            ReDim nullArr(1 To n, 1 To 1)
            For i = 1 To n
                nullArr(i, 1) = 0
            Next i
            .Values = nullArr
        Else
            .Values = OneColTwoDimArray(chrtDataArr, arrColIndex)
        End If
    End With

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub
```

```
Option Explicit
    Private Declare PtrSafe Function Beep Lib "kernel32" (ByVal dwFreq As Long, ByVal dwDuration As
Long) As Long

Public Sub PlayBeep()
    Beep 1, 1
    Beep 700, 250
    Beep 800, 250
    Beep 600, 250
End Sub

Public Function IsFolder(folderPath As String) As Boolean
    'On Error GoTo ErrorHandler
    Dim folderObj As Object

    Set folderObj = CreateObject("Scripting.FileSystemObject")
    IsFolder = folderObj.folderExists(folderPath)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function IsFile(fileAddress As String) As Boolean
    'On Error GoTo ErrorHandler

    IsFile = (Dir(fileAddress) <> "")

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function IsFormEmpty(fm As UserForm) As Boolean
    'On Error GoTo ErrorHandler
    Dim sCount As Integer, tCount As Integer, fCount As Integer
    Dim ctl As MSForms.Control
    IsFormEmpty = True

    For Each ctl In fm.Controls
        Select Case TypeName(ctl)
            Case "TextBox"
                If Trim(ctl.Text) = "" And Not ctl.Name = "BoxNotes" Then Exit Function
                If Trim(ctl.Value) = "" And Not ctl.Name = "BoxNotes" Then Exit Function
            Case "ComboBox", "ListBox"
                If ctl.ListIndex = -1 Then Exit Function
            Case "CheckBox", "OptionButton", "ToggleButton"
                sCount = sCount + 1
                If ctl.Value = False Then fCount = fCount + 1
        End Select
    Next ctl
    If sCount <> 0 And sCount = fCount Then Exit Function

    IsFormEmpty = False

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function SelectItem(ByVal listItemsArray As Variant) As Integer
```

```
    'On Error GoTo ErrorHandler
    Dim fm As Object

    Set fm = VBA.UserForms.Add("UF_List")
    fm.ItemsList = listItemsArray
    fm.Show vbModal
    SelectItem = fm.SelectedItemIndex + 1

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function OverWriteArray(ByRef owDataArr, ByRef owColArr As Variant, ByVal colIndex As Integer)
As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, owLastRow As Long, tempOWColArr As Variant

    tempOWColArr = TransposeArray(owColArr)
    owLastRow = UBound(tempOWColArr, 1)

    For i = 1 To owLastRow
        owDataArr(i, colIndex) = tempOWColArr(i, 1)
    Next i

    OverWriteArray = owDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function OneColTwoDimArray(ByRef twoDimArr As Variant, ByVal arrColIndex As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim arrLength As Long, i As Long, tempArr() As Double

    arrLength = UBound(twoDimArr, 1)
    ReDim tempArr(1 To arrLength, 1 To 1)

    For i = 1 To arrLength
        tempArr(i, 1) = twoDimArr(i, arrColIndex)
    Next i

    OneColTwoDimArray = tempArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function TransposeArray(ByRef transDataArr As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, j As Long, dimNum As Long
    Dim tempArr() As Variant

    dimNum = 0
    On Error Resume Next
    dimNum = UBound(transDataArr, 2)
    On Error GoTo 0

    If dimNum = 0 Then
        ReDim tempArr(1 To UBound(transDataArr), 1 To 1)
```

```
        j = IIf(LBound(transDataArr) = 0, 1, 0)
        For i = 1 To UBound(transDataArr) + j
            tempArr(i, 1) = transDataArr(i - j)
        Next i
        TransposeArray = tempArr
    ElseIf dimNum > 1 Then
        ReDim tempArr(1 To UBound(transDataArr, 2), 1 To 1)
        For i = 1 To dimNum
            tempArr(i, 1) = transDataArr(1, i)
        Next i
        TransposeArray = tempArr
    Else
        TransposeArray = transDataArr
    End If

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function ResizeArrayShrink(ByRef resDataArr As Variant, ByVal startColIndex As Integer, ByVal
endColIndex As Integer, _
    ByVal startRowIndex As Long, ByVal endRowIndex As Long) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Integer, oldLastCol As Integer, newLastCol As Integer
    Dim j As Long, oldLastRow As Long, newLastRow As Long
    Dim tempArr As Variant

    oldLastRow = UBound(resDataArr, 1)
    oldLastCol = UBound(resDataArr, 2)
    newLastRow = endRowIndex - startRowIndex + 1
    newLastCol = endColIndex - startColIndex + 1
    ReDim tempArr(1 To newLastRow, 1 To newLastCol)

    For i = 1 To newLastCol
        For j = 1 To newLastRow
            tempArr(j, i) = resDataArr(startRowIndex + j - 1, startColIndex + i - 1)
        Next j
    Next i

    ResizeArrayShrink = tempArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function OneDimArray(ByRef twoDimArr As Variant, ByVal arrColIndex As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim arrLength As Long, i As Long
    Dim tempArr() As Double

    arrLength = UBound(twoDimArr, 1)
    ReDim tempArr(1 To arrLength)

    For i = 1 To arrLength
        tempArr(i) = twoDimArr(i, arrColIndex)
    Next i

    OneDimArray = tempArr

    Exit Function
```

```
ErrorHandler:
    Call ErrorHandler
End Function

Public Function GetWinArr(ByRef winDataArr As Variant, ByVal centerIndex As Long, ByVal winWinSize As
Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim halfWinSize As Integer, i As Long, tempArr() As Double

    If winWinSize Mod 2 = 0 Then winWinSize = winWinSize + 1

    halfWinSize = winWinSize \ 2
    ReDim tempArr(1 To winWinSize)

    For i = 1 To winWinSize
        tempArr(i) = winDataArr(centerIndex - halfWinSize + i - 1)
    Next i

    GetWinArr = tempArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function ResizeArrayExpand(ByRef resDataArr As Variant, ByRef expDataArr As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim resLastCol As Integer, expLastCol As Integer, lastCol As Integer, resLastRow As Long,
expLastRow As Long, lastRow As Long
    Dim i As Long, j As Long, tempExpArr As Variant, tempArr() As Double

    tempExpArr = TransposeArray(expDataArr)
    resLastRow = UBound(resDataArr, 1)
    resLastCol = UBound(resDataArr, 2)
    expLastRow = UBound(tempExpArr, 1)
    expLastCol = UBound(tempExpArr, 2)
    lastRow = IIf(resLastRow >= expLastRow, resLastRow, expLastRow)
    lastCol = resLastCol + expLastCol

    ReDim tempArr(1 To lastRow, 1 To lastCol)

    For i = 1 To resLastCol
        For j = 1 To resLastRow
            tempArr(j, i) = resDataArr(j, i)
        Next j
    Next i

    For i = 1 To expLastCol
        For j = 1 To expLastRow
            tempArr(j, resLastCol + i) = tempExpArr(j, i)
        Next j
    Next i

    ResizeArrayExpand = tempArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function FillArray(ByRef fillDataArr As Variant, ByVal startRow As Long, ByVal endRow As Long,
ByVal fillValue As Double) As Variant
```

```
    'On Error GoTo ErrorHandler
    Dim i As Long, lastRow As Long, tempArr() As Double

    lastRow = UBound(fillDataArr)
    ReDim tempArr(1 To lastRow)

    For i = 1 To startRow - 1
        tempArr(i) = fillDataArr(i)
    Next i

    For i = startRow To endRow
        tempArr(i) = fillValue
    Next i

    For i = endRow + 1 To lastRow
        tempArr(i) = fillDataArr(i)
    Next i

    FillArray = tempArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function FindInArray(ByRef iDataArr As Variant, ByVal arrElement As Double) As Long
    'On Error GoTo ErrorHandler
    Dim i As Long

    FindInArray = 0

    For i = 1 To UBound(iDataArr)
        If iDataArr(i) = arrElement Then
            FindInArray = i
            Exit Function
        End If
    Next i

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Sub SortArray(ByRef sortDataArr() As Double, ByVal low As Long, ByVal high As Long)
    'On Error GoTo ErrorHandler
    Dim pivot As Double, temp As Double
    Dim i As Long, j As Long

    i = low
    j = high
    pivot = sortDataArr((low + high) \ 2)

    Do While i <= j
        Do While sortDataArr(i) < pivot
            i = i + 1
        Loop

        Do While sortDataArr(j) > pivot
            j = j - 1
        Loop

        If i <= j Then
```

```
            temp = sortDataArr(i)
            sortDataArr(i) = sortDataArr(j)
            sortDataArr(j) = temp
            i = i + 1
            j = j - 1
        End If
    Loop

    If low < j Then SortArray sortDataArr, low, j
    If i < high Then SortArray sortDataArr, i, high

    Exit Sub
ErrorHandler:
    Call ErrorHandler
End Sub

Public Function CalcMedian(ByRef mDataArr As Variant) As Double
    'On Error GoTo ErrorHandler
    Dim sortDataArr() As Double
    Dim i As Long, j As Long, n As Long, middle As Long

    n = UBound(mDataArr)
    ReDim sortDataArr(1 To n)

    j = 0
    For i = 1 To n
        If Not IsEmpty(mDataArr(i)) Then
            j = j + 1
            sortDataArr(j) = mDataArr(i)
        End If
    Next i

    If j > 2 Then
        Call SortArray(sortDataArr, LBound(sortDataArr), UBound(sortDataArr))
    Else
        CalcMedian = CalcAverage(sortDataArr)
        Exit Function
    End If

    n = UBound(sortDataArr) - LBound(sortDataArr) + 1
    middle = LBound(sortDataArr) + (n \ 2)

    If n Mod 2 = 0 Then
        CalcMedian = (sortDataArr(middle) + sortDataArr(middle - 1)) / 2
    Else
        CalcMedian = sortDataArr(middle)
    End If

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function CalcAverage(ByRef avgDataArr As Variant) As Double
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, subSum As Double

    n = UBound(avgDataArr) - LBound(avgDataArr) + 1

    For i = LBound(avgDataArr) To UBound(avgDataArr)
        subSum = subSum + avgDataArr(i)
    Next i
```

```
    CalcAverage = subSum / n

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function MovingAvg(ByRef normDataArr As Variant, ByVal normWinSize As Integer) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, halfWinSize As Integer

    n = UBound(normDataArr)
    halfWinSize = normWinSize \ 2
    For i = halfWinSize + 1 To n - halfWinSize
        normDataArr(i) = CalcAverage(GetWinArr(normDataArr, i, normWinSize))
    Next i

    MovingAvg = normDataArr

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function CalcVariance(ByRef vDataArr As Variant) As Double
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long, mean As Double, sumSqDiff As Double

    n = UBound(vDataArr) - LBound(vDataArr) + 1
    mean = CalcAverage(vDataArr)

    For i = LBound(vDataArr) To UBound(vDataArr)
        sumSqDiff = sumSqDiff + (vDataArr(i) - mean) ^ 2
    Next i

    CalcVariance = Sqr(sumSqDiff / n)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function CalcRegress(ByRef dyWinArr As Variant) As Variant
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long
    Dim xVal As Double, yVal As Double, xSum As Double, ySum As Double, xySum As Double, xxSum As
Double, _
    Slope As Double, intercept As Double, yPred As Double, varSum As Double, regVals() As Double

    n = UBound(dyWinArr)
    For i = 1 To n
        xVal = i
        yVal = dyWinArr(i)
        xSum = xSum + xVal
        ySum = ySum + yVal
        xySum = xySum + xVal * yVal
        xxSum = xxSum + xVal ^ 2
    Next i

    If n * xxSum - xSum ^ 2 = 0 Then
        Slope = Settings.AppSettings("Max Horizontal Angle")
```

```
    Else
        Slope = (n * xySum - xSum * ySum) / (n * xxSum - xSum ^ 2)
    End If

    intercept = (ySum - Slope * xSum) / n

    ReDim regVals(1 To 2)
    xVal = n - n \ 2
    regVals(2) = (Slope * xVal) + intercept

    For i = 1 To n
        xVal = i
        yVal = dyWinArr(i)
        yPred = (Slope * xVal) + intercept
        'varSum = varSum + (yVal - yPred) ^ 2
        varSum = varSum + Abs(yVal - yPred)
    Next i

    'regVals(1) = Sqr(varSum / n)
    regVals(1) = varSum / n

    CalcRegress = regVals

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function CalcSlope(ByRef slWinArr As Variant) As Double
    'On Error GoTo ErrorHandler
    Dim i As Long, n As Long
    Dim xVal As Double, yVal As Double, xSum As Double, ySum As Double, xySum As Double, xxSum As
Double

    n = UBound(slWinArr)
    For i = 1 To n
        xVal = i
        yVal = slWinArr(i)
        xSum = xSum + xVal
        ySum = ySum + yVal
        xySum = xySum + xVal * yVal
        xxSum = xxSum + xVal ^ 2
    Next i

    If Not n * xxSum - xSum ^ 2 = 0 Then CalcSlope = (n * xySum - xSum * ySum) / (n * xxSum - xSum ^
2)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function FindStartRow(ByRef stDataArr As Variant, ByVal wsName As String, ByVal rvThreshold As
Double, ByVal avgThreshold As Double) As Long
    'On Error GoTo ErrorHandler
    Dim i As Integer, j As Integer, fStartRow(1 To 2) As Long

    Select Case Right(wsName, 3)
        Case "tal", "ght", "eft"
            j = 1
        Case "cal", " Up", "own"
            j = 3
```

```
    End Select

    For i = 1 To 2
        fStartRow(i) = FindDeflect(OneDimArray(stDataArr, i + j - 1), avgThreshold)
    Next i

    FindStartRow = (fStartRow(1) + fStartRow(2)) \ IIf(fStartRow(1) * fStartRow(2) = 0, 1, 2)

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function FindDeflect(ByRef dDataArr As Variant, ByVal avgThreshold As Double) As Variant
    'On Error GoTo ErrorHandler
    Dim freqRowCount As Integer, dWinSize As Integer, tempDeflect(1 To 2) As Long, i As Long, j As
Long, n As Long
    Dim dWinAvg() As Double, dWinPred() As Double, dWinRV() As Double, dBaseAvg(1 To 2) As Double,
befDiff As Double, aftDiff As Double
    Dim tempRegress As Variant

    tempDeflect(1) = 0
    tempDeflect(2) = 0
    freqRowCount = Settings.ProcessSettings("Frequency")
    dWinSize = freqRowCount \ 10
    If dWinSize Mod 2 = 0 Then dWinSize = dWinSize + 1

    n = UBound(dDataArr) - dWinSize
    ReDim dWinPred(1 To n)
    ReDim dWinRV(1 To n)
    For i = 1 To n
        tempRegress = CalcRegress(GetWinArr(dDataArr, i + dWinSize \ 2 + 1, dWinSize))
        dWinRV(i) = tempRegress(1)
        dWinPred(i) = tempRegress(2)
    Next i

    n = UBound(dWinPred) - dWinSize
    ReDim dWinAvg(1 To n)
    For i = 1 To n
        dWinAvg(i) = CalcAverage(GetWinArr(dWinPred, i + dWinSize \ 2 + 1, dWinSize))
    Next i

    dBaseAvg(1) = 22 'CalcAverage(GetWinArr(dWinAvg, freqRowCount \ 2 + 1, freqRowCount))
    dBaseAvg(2) = 22 'CalcAverage(GetWinArr(dWinAvg, n - freqRowCount \ 2, freqRowCount))

    n = UBound(dWinAvg) - dWinSize
    For i = 1 To n - 10 * dWinSize
        j = 1
        Do While j < 10
            If Abs(dWinAvg(i)) + avgThreshold * j > Abs(dWinAvg(i + j * dWinSize)) Then Exit Do
            j = j + 1
        Loop
        If j = 10 Then
            j = 1
            befDiff = 0
            Do While j < dWinSize
                If Abs(dWinAvg(i)) > Abs(dWinAvg(i + j)) Then Exit Do
                aftDiff = dWinRV(i + j)
                If aftDiff > befDiff Then
                    befDiff = aftDiff
                    tempDeflect(1) = i + j
                End If
```

```
                j = j + 1
            Loop
            If j = dWinSize Then
                tempDeflect(1) = tempDeflect(1) + dWinSize \ 2
                Exit For
            End If
        End If
    Next i

    For i = n To 1 + 10 * dWinSize Step -1
        j = 1
        Do While j < 10
            If Abs(dWinAvg(i)) + avgThreshold * j > Abs(dWinAvg(i - j * dWinSize)) Then Exit Do
            j = j + 1
        Loop
        If j = 10 Then
            j = 1
            befDiff = 0
            Do While j < dWinSize
                If Abs(dWinAvg(i)) > Abs(dWinAvg(i - j)) Then Exit Do
                aftDiff = dWinRV(i - j)
                If aftDiff > befDiff Then
                    befDiff = aftDiff
                    tempDeflect(2) = i - j
                End If
                j = j + 1
            Loop
            If j = dWinSize Then
                tempDeflect(2) = tempDeflect(2) + dWinSize \ 2
                Exit For
            End If
        End If
    Next i

    FindDeflect = tempDeflect

    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function

Public Function FindPeak(ByRef peDataArr As Variant, ByVal maxY As Double) As Long
    'On Error GoTo ErrorHandler
    Dim n As Long, i As Long, maxX As Long, peak As Long, dist As Double, tempDist As Double

    n = UBound(peDataArr)
    dist = 2 * n ^ 2
    maxX = n \ 2
    maxY = maxY * 1.5

    If maxY < 0 Then maxY = maxY * -1

    For i = 1 To n
        tempDist = Sqr((maxX - i) ^ 2 + ((maxY - peDataArr(i)) * n / (maxY * 2)) ^ 2)
        If tempDist < dist Then
            dist = tempDist
            peak = i
        End If
    Next i

    FindPeak = peak
```

```
    Exit Function
ErrorHandler:
    Call ErrorHandler
End Function
```

Appendix 2: Consent Form

CONSENT FORM TO PARTICIPATE IN "DETS" STUDY

You are being asked to take part in a research study. Please read the information about the study presented in this form. The form includes details on study's risks and benefits that you should know before you decide if you would like to take part. You should take as much time as you need to make your decision. You should ask the investigator to explain anything that you do not understand and make sure that all your questions have been answered before signing this consent form. Before you make your decision, feel free to talk about this study with anyone you wish, including your friends, family, and doctors.

Your participation in this research is entirely voluntary. It is your choice whether to participate or not. Whether you choose to participate or not, all the services you receive at this clinic/hospital will continue and nothing will change. Your decision not to participate will not have any negative effect on you or your medical care.

This Informed Consent Form has two parts:

Part 1: Information Sheet (to share information about the research with you)

Part 2: Certificate of Consent (for signatures if you agree to take part)

TITLE

Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders (DETS)

INVESTIGATOR

Dr. Hadi Karimi, MD, MSc

SUPERVISOR

Prof. Anandan Moodley, MBChB (Natal), M.HPE (UFS), FCP Neurol (SA), FEBN (EU), PhD (UKZN), F.R.C.P.(C)

INSTITUTIONAL AFFILIATIONS

Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal, Nelson Mandela School of Medicine, Durban, South Africa

CONSENT FORM TO PARTICIPATE IN "DETS" STUDY

PART I: INFORMATION SHEET

We are conducting a research study on the use of automated eye tracking as an additional test for neurological disorders. The purpose of this study is to evaluate the accuracy and feasibility of this technology in diagnosing of various conditions that affect the brain and nervous system with a more efficient, cost effective, and non-invasive method. You are invited to participate in this study because you either have been diagnosed with a neurological disorder or you are a healthy volunteer. Your participation is voluntary, and you can withdraw from the study at any time without any consequences.

If you agree to participate, we will ask you a few questions about your general health and collect data from your records to assess if you are eligible to participate in this study. We will collect information about your present and past health, as well as the names and reasons for any medications you are currently taking. The data collected from the eye tracking tests will be stored in a secure database and analyzed by our research team.

To participate in this research, you must allow the study team to use your health information. If you do not want us to use your protected health information, you may not participate in this study. Information about you that will be collected during the research will be put away and no-one but the researchers will be able to see it. Any information about you will have a number on it instead of your name. Only the researchers will know what your number is, and we will lock that information up with a lock and key. It will not be shared with or given to anyone except the research ethics review board, your clinician, or people you designate. Patients usually have a right to access their medical records. However, you may not be allowed to see or copy certain information that is related to this research study.

If we assess you are an eligible candidate to participate in this study, you will be asked to undergo an eye tracking test using a device that records your eye movements while you are comfortably sitting and looking at different presentations on a screen. You will be asked to follow a target on the screen. Clear directions will be given to you at the time of test. The tests will take about 30 minutes. No further tests or your involvements are required. We will not be taking any bloods or doing any extra tests for the sake of this research. There will be no cost to you. Also, you will not be paid for participating in this research study. You will not be given any money, gifts, or reimbursement for any costs you may endure for taking part in this research.

We do not expect any adverse effects from this test. However, the possible risks of participating in this study is that you may experience some discomfort or fatigue during the eye tracking tests. In such case, we would stop the test immediately, and we expect such effects to resolve soon after. While the possibility of this happening is very low, you should still be aware of the possibility and if something unexpected happens, you will not be compensated in any shape or form, and we do not provide you with any additional medical insurance. However, the investigator will be available to assess and assist you in such events.

You do not have to take part in this research if you do not wish to do so, and refusing to participate will not affect your treatment at this clinic/hospital in any way. You will still have all the benefits that you

would otherwise have at this clinic/hospital. You may stop participating in the research at any time that you wish without losing any of your rights as a patient here.

If you participate in this research, there may not be any benefit for you, but your participation by contributing to the advancement of knowledge and practice in the field of neurology is likely to benefit society in future.

If you have any questions, you may ask them now or later, or during the test.

Thank you for your interest and cooperation.

PART II: CERTIFICATE OF CONSENT

I have read (or it has been read to me) and I understood the foregoing information. I have had the opportunity to ask questions about it and any questions that I have asked, have been answered to my satisfaction. I consent voluntarily to take part as a participant in this research and I agree to the use of my information as described in this form.

Name of Participant	Signature of Participant	Date

I have witnessed the accurate reading of the consent form to the potential participant, and the individual has had the opportunity to ask questions. I confirm that the individual has given consent freely.

Name of Participant	Signature of Participant	Date

I have accurately read out the information sheet to the potential participant, and to the best of my ability made sure that the participant understands. I confirm that the participant was given an opportunity to ask questions about the study, and all the questions asked by the participant have been answered correctly and to the best of my ability. I confirm that the individual has not been coerced into giving consent, and the consent has been given freely and voluntarily.

Name of Participant	Signature of Participant	Date

IFOMU LOKUVUMA UKUHLANGANYELA OCWANINGWENI LWE-"DETS"

Uyacelwa ukuthi uhlanganyele ocwaningweni. Sicela ufunde ulwazi oluqondene nocwaningo olukhonjiswe kuleli fomu. Le fomu iqukethe imininingwane mayelana nobungozi nezinzuzo zocwaningo okufanele uyazi ngaphambi kokuba unqume ukuthi ufuna yini ukuhlanganyela. Kufanele uthathe isikhathi esidingekayo ukwenza isinqumo sakho. Kufanele ubuze umcwaningi ukuthi achaze noma yini ongayiqondi futhi uqinisekise ukuthi yonke imibuzo yakho iphenduliwe ngaphambi kokusayina leli fomu lokuvuma. Ngaphambi kokwenza isinqumo sakho, zizwe ukhululekile ukuxoxa ngocwaningo lolu nanoma ubani ongathanda ukukhuluma naye, kuhlanganisa nabangane bakho, umndeni wakho kanye nodokotela bakho.

Ukuthatha kwakho kulo cwaningo kungenxa yokuzikhethela kwakho ngokuphelele. Kuyisinqumo sakho ukuthi ufuna ukungena noma cha. Noma ukhetha ukungena noma cha, zonke izinsiza ozitholayo kulesi kliniki/isibhedlela zizoqhubeka futhi akukho okuzoshintsha. Isinqumo sakho sokungabandakanyi ngeke sibe nomthelela ongemuhle kuwe noma ekunakekelweni kwakho kwezokwelapha.

ISIHLOKO

Ukusetshenziswa Kokulandelela Amehlo Okuzenzakalelayo Njengesivivinyo Esisizayo Ezinkingeni Zezifo Zezinzwa (DETS)

UMCWANINGI

UDkt. Hadi Karimi MD MSc

UMHOLI

UProf. Anandan Moodley MBChB (Natal) M.HPE (UFS) FCP Neurol (SA) FEBN (EU) PhD (UKZN) F.R.C.P.(C)

UKUHLOMELISA KWESIKHUNGO

UMnyango Wezifo Zezinzwa, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal, Nelson Mandela School of Medicine, Durban, South Africa

IFOMU LOKUVUMA UKUHLANGANYELA OCWANINGWENI LWE-"DETS"

INGXENYE I: ISHIDI LELWAZI

Senza ucwaningo ngokusetshenziswa kokulandelela amehlo okuzenzakalelayo njengesivivinyo esengeziwe ezinkingeni zezifo zezinzwa. Injongo yalo cwaningo wukuhlola ukunemba nokwenzeka kwale teknolojia ekuhlolweni kwezimo ezihlukahlukene ezithinta ubuchopho kanye nesistimu yezinzwa ngezindlela ezingabizi, ezisebenza kahle futhi ezingaphikisiyo. Umema ukuba ube yingxenye yalolu cwaningo ngoba usuke utholwe unezifo zezinzwa noma ungumuntu onempilo. Ukubamba kwakho iqhaza kungenxa yokuzikhethela futhi ungahle uhlehle noma yinini ngaphandle kwemiphumela.

Uma uvuma ukuhlanganyela, sizokubuza imibuzo embalwa mayelana nempilo yakho ejwayelekile futhi siqoqe idatha evela kumarekhodi akho ukuze sihlole ukuthi uyafaneleka yini ukuhlanganyela kulo cwaningo. Sizokuqoqela ulwazi mayelana nempilo yakho yamanje neyangaphambili kanye namagama nezizathu zemithi oyisebenzisayo okwamanje. Imininingwane eqoqwe ekuhloleni amehlo izogcinwa kudathabheyisi evikelekile futhi izohlaziywa ithimba lethu locwaningo.

Ukuze ube yingxenye yocwaningo lolu, kufanele uvumele ithimba locwaningo ukuthi lisebenzise ulwazi lwakho lwezempilo. Uma ungafuni ukuthi sisebenzise ulwazi lwakho lwezempilo oluvikelekile, ngeke uhlanganyele kulo cwaningo. Ulwazi mayelana nawe oluqoqwe ngesikhathi socwaningo luzovalelwa futhi akukho muntu ngaphandle kwabacwaningi ozolubona. Noma yiluphi ulwazi mayelana nawe luzoba nenombolo kunokuba igama lakho. Abacwaningi kuphela abazokwazi ukuthi iyiphi inombolo yakho futhi sizolukhiya ngendlu evalekile. Ngeke yabelane noma inikwe muntu ngaphandle kwebhodi locwaningo lwezethika, udokotela wakho noma abantu obaqokayo. Iziguli ngokuvamile zinelungelo lokuthola amarekhodi azo ezokwelapha. Nokho, ungase ungavunyelwe ukubona noma ukukopisha ulwazi oluthile oluhlobene nalo cwaningo.

Uma sihlola ukuthi uyafaneleka ukubamba iqhaza kulo cwaningo, uzocelwa ukuthi uthole ukuhlolwa kwamehlo usebenzisa umshini oqopha ukunyakaza kwamehlo akho ngenkathi uhlezi ngokunethezeka futhi ubuka izethulo ezihlukahlukene esikrinini. Uzocelwa ukuthi ulandele isikhombi esikrinini. Imiyalelo ecacile izonikezwa ngesikhathi sokuhlolwa. Ukuhlolwa kuzothatha cishe imizuzu engama-30. Akuwona amanye ukuhlolwa noma ukugxila kwakho okudingekayo. Ngeke sithathe noma yiliphi igazi noma senze ukuhlolwa okungeziwe ngenxa yalolu cwaningo. Ngeke kube khona izindleko kuwe. Futhi awuyikukhokhelwa ngokubamba iqhaza kulolu cwaningo. Awuyikuthola imali, izipho noma izimbuyiselo zanoma yiziphi izindleko ongaba nazo ngokuthatha iqhaza kulolu cwaningo.

Asilindele ukuthi kube nemiphumela emibi evela kulokhu kuhlolwa. Kodwa-ke, ubungozi obungaba khona bokuthatha iqhaza kulo cwaningo ukuthi ungase ube nokungakhululeki noma ukhathele ngesikhathi sokuhlolwa kwamehlo. Uma kunjalo, sizomisa ukuhlolwa ngokushesha futhi silindele ukuthi leyo miphumela iphele ngokushesha. Nakuba amathuba okuthi lokhu kwenzeke ephansi kakhulu, kufanele usaziswe ukuthi kungenzeka futhi uma kwenzeka okungakuhlelwanga ngeke ube nenhlawulo ngendlela ethile futhi asikunikezi umshuwalense wokwelashwa owengeziwe. Nokho, umcwaningi uzoba khona ukuze ahlole futhi asize uma kwenzeka lokhu.

Awudingi ukuhlanganyela kulo cwaningo uma ungafuni futhi ukwenqaba ukuthatha iqhaza ngeke kube nomphumela wokwelashwa kwakho kulesi kliniki/isibhedlela nganoma iyiphi indlela. Uzoqhubeka nokuthola zonke izinzuzo obuzothola kulesi kliniki/isibhedlela. Ungayeka

ukuhlanganyela ocwaningweni nganoma yisiphi isikhathi ofisa ngaphandle kokulahlekelwa amalungelo akho njengesiguli lapha.

Uma ubamba iqhaza kulo cwaningo, kungenzeka kungabi nanzuzo kuwe kodwa ukuthatha kwakho iqhaza ngokufaka isandla ekuthuthukiseni ulwazi nokusebenza emkhakheni wezifo zezinzwa kungenzeka kube usizo emphakathini esikhathini esizayo.

Uma unemibuzo ungayibuza manje noma kamuva noma ngesikhathi sokuhlolwa.

Siyabonga ngokuba nentshisekelo nokubambisana kwakho.

INGXENYE II: ISITIFIKETHI SEMVUME

Ngifunde kahle ikhasi lolwazi kumhlanganyeli ongaba khona futhi ngokusemandleni ami ngiqinisekise ukuthi umhlanganyeli uyaqonda. Ngiyavuma ukuthi umhlanganyeli unikezwe ithuba lokubuza imibuzo mayelana nocwaningo futhi yonke imibuzo ebuziwe ngumhlanganyeli iphendulwe kahle futhi ngokusemandleni ami. Ngiyavuma ukuthi umuntu akazange aphoqelelwe ekunikezeni imvume futhi imvume inikezwe ngokukhululekile futhi ngokuzithandela.

Igama Lomhlanganyeli	Isiginesha Somhlanganyeli	Usuku

Ngifakazile ukufunda kahle kwalesi sikhulu kwemvume kumhlanganyeli ongaba khona futhi umuntu unikezwe ithuba lokubuza imibuzo. Ngiyavuma ukuthi umuntu unikeze imvume ngokukhululekile.

Igama Lomhlanganyeli	Isiginesha Somhlanganyeli	Usuku

Ngifunde (noma kuye kwafundwa kimi) futhi ngiyaqonda ulwazi olungenhla. Ngibe nethuba lokubuza imibuzo ngalokhu futhi yonke imibuzo ebengiyibuza iphenduliwe ngendlela engijabulisayo. Ngiyavuma ngokukhululekile ukuthatha iqhaza njengomhlanganyeli kulo cwaningo futhi ngiyavuma ukusetshenziswa kolwazi lwami njengoba kuchazwe kuleli fomu.

Igama Lomhlanganyeli	Isiginesha Somhlanganyeli	Usuku

Appendix 3: Ethics Approvals

UNIVERSITY OF KWAZULU-NATAL
INYUVESI YAKWAZULU-NATALI

21 June 2024

Dr Hadi Karimi (218086575)
School of Clinical Medicine
Medical School

Dear Dr Karimi,

Protocol reference number: BREC/00006696/2024
Project title: Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders
Degree: MMed

EXPEDITED APPLICATION: APPROVAL LETTER

A sub-committee of the Biomedical Research Ethics Committee has considered and noted your application.

The conditions have been met and the study is given full ethics approval and may begin as from 21 June 2024. Please ensure that any outstanding site permissions are obtained and forwarded to BREC for approval before commencing research at a site.

This approval is valid for one year from 21 June 2024. To ensure uninterrupted approval of this study beyond the approval expiry date, an application for recertification must be submitted to BREC on RIG on the appropriate BREC form 2-3 months before the expiry date.

Any amendments to this study, unless urgently required to ensure safety of participants, must be approved by BREC prior to implementation.

Your acceptance of this approval denotes your compliance with South African National Research Ethics Guidelines (2024), South African National Good Clinical Practice Guidelines (2020) (if applicable) and with UKZN BREC ethics requirements as contained in the UKZN BREC Terms of Reference and Standard Operating Procedures, all available at http://research.ukzn.ac.za/Research-Ethics/Biomedical-Research-Ethics.aspx.

BREC is registered with the South African National Health Research Ethics Council (REC-290408-009). BREC has US Office for Human Research Protections (OHRP) Federal-wide Assurance (FWA 678).

The sub-committee's decision will be noted by a full Committee at its next meeting taking place on 09 July 2024.

Yours sincerely,

[redacted]
Prof S Singh
Chair: Biomedical Research Ethics Committee

Biomedical Research Ethics Committee
Chair: Professor S Singh
UKZN Research Ethics Office Westville Campus, Govan Mbeki Building
Postal Address: Private Bag X54001, Durban 4000
Email: BREC@ukzn.ac.za
Website: http://research.ukzn.ac.za/Research-Ethics/Biomedical-Research-Ethics.aspx

KWAZULU-NATAL PROVINCE
HEALTH
REPUBLIC OF SOUTH AFRICA

DIRECTORATE:

Physical Address: 330 Langalibalele Street, Pietermaritzburg
Postal Address: Private Bag X9051
Tel: 033 395 2805/ 3169/ 3123 Fax: 033 394 3782
Email:

Health Research & Knowledge Management

NHRD Ref: KZ_202403_020

Dear Dr H Karim
(UKZN)

Approval of research

1. The research proposal titled **'Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders'** was reviewed by the KwaZulu-Natal Department of Health (KZN-DoH).

   The proposal is hereby **approved** for research to be undertaken at Inkosi Albert Luthuli Central hospital.

2. You are requested to take note of the following:
   a. ***Kindly liaise with the facility manager BEFORE your research begins.***
      This is to ensure that conditions in the facility are conducive to the conduct of your research. These include, but are not limited to, an assurance that the numbers of patients attending the facility are sufficient to support your sample size requirements, and that the space and physical infrastructure of the facility can accommodate the research team and any additional equipment required for the research.
   b. *All research conducted in KwaZulu-Natal must comply with government regulations relating to Covid-19. These include but are not limited to: regulations concerning social distancing, the wearing of personal protective equipment, and limitations on meetings and social gatherings.*
   c. *Please ensure that you provide your letter of ethics re-certification to this unit when the current approval expires.*
   d. *Provide an interim progress report and final report (electronic and hard copies) when your research is complete to **HEALTH RESEARCH AND KNOWLEDGE MANAGEMENT, 10-102, PRIVATE BAG X9051, PIETERMARITZBURG, 3200** and e-mail an electronic copy to hrkm@kznhealth.gov.za*
   e. *Please note that the Department of Health shall not be held liable for any injury that occurs as a result of this study.*

For any additional information please contact Dr. G Shezi on 033-395 3189.

Yours Sincerely

[redacted]

Dr E Lutge
Chairperson, Provincial Health Research Committee
Date: 07/05/2024.

KWAZULU-NATAL PROVINCE
HEALTH
REPUBLIC OF SOUTH AFRICA

DIRECTORATE:

NKOSI ALBERT LUTHULI CENTRAL HOSPITAL
OFFICE OF THE MEDICAL MANAGER
Private Bag X03, Mayville, 4058
300 Vusi Mzimela (Bellair) Road, Mayville,4091
Tel: 031 240 1059 Fax: 031 240 1005 Email: Ursula.john@ialch.co.za

Reference: BREC 00006696/2024
Enquiries: Medical Management

19 April 2024

Dr H Karimi (218086575)
School of Clinical Medicine
Medical School

Dear Dr Karimi

RE: PERMISSION TO CONDUCT RESEARCH AT IALCH

I have pleasure in informing you that permission has been granted to you by the Medical Manager to conduct research on: **Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders.**

Kindly take note of the following information before you continue:

1. Please ensure that you adhere to all the policies, procedures, protocols and guidelines of the Department of Health with regards to this research.
2. This research will only commence once this office has received confirmation from the Provincial Health Research Committee in the KZN Department of Health.
3. Kindly ensure that this office is informed before you commence your research.
4. The hospital will not provide any resources for this research.
5. You will be expected to provide feedback once your research is complete to the Medical Manager.

Yours faithfully

[redacted]

.....

Dr S Singh
Clinical Care Manager

Appendix 4: Research Protocol

Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders

RESEARCH PROTOCOL

TITLE

Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders.

SHORT TITLE

Diagnostic Eye Tracking Study (DETS)

INVESTIGATOR

Dr. Hadi Karimi, MD, MSc
Supernumerary Registrar
218086575@stu.ukzn.ac.za

SUPERVISOR

Prof. Anandan Moodley, MBChB (Natal), M.HPE (UFS), FCP Neurol (SA), FEBN (EU), PhD (UKZN), F.R.C.P.(C)
Department of Neurology, University of KwaZulu-Natal, Nelson Mandela School of Medicine
Moodleya20@ukzn.ac.za

INSTITUTIONAL AFFILIATIONS

Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal, Nelson Mandela School of Medicine, Durban, South Africa

BACKGROUND AND RATIONALE

Neurological diseases are disorders that affect the brain, spinal cord, or peripheral nerves, and are a major cause of disability and mortality worldwide, affecting millions of people and imposing a huge burden on the health care system [1, 2]. Neurological diseases can be classified into different categories, such as neurodegenerative diseases (e.g., Alzheimer's disease, Parkinson's disease, Amyotrophic Lateral Sclerosis), demyelinating diseases (e.g., multiple sclerosis), cerebrovascular diseases (e.g., stroke), genetic (e.g., Huntington's disease, Cerebellar diseases, Mitochondrial disorders), neuromuscular diseases (e.g., Myasthenia Gravis), or immune-mediated diseases (e.g., Inflammatory Demyelination Polyneuropathy) [2, 3].

The diagnosis of neurological diseases is often challenging and complex, as it requires a combination of clinical history, physical examination, as well as diagnostics such as laboratory tests (blood, CSF), Neurophysiology studies and neuroimaging. However, these diagnostic methods have some limitations, such as being invasive, costly, time-consuming, or not widely available, and sometimes not specific. Moreover, some neurological diseases may have overlapping or nonspecific findings, making it difficult to distinguish them from each other or from normal aging [4, 5]. Therefore, there is a need for more efficient, reliable, and accessible methods to screen and diagnose neurological diseases at an early stage before irreversible damage occurs, and to monitor their progression and response to treatment.

Eye tracking is a non-invasive technique that measures eye movements and gaze patterns while subjects view visual stimuli. Eye tracking reveals information about the functional status of the brain regions and pathways involved in visual processing, eye movement control and the underlying neuronal circuitry. To date, eye tracking has been mostly used to study various aspects of human brain cognitive function, such as attention, decision-making, and behavior [6-9]. However recently, eye tracking has been proposed as a potential diagnostic tool for neurological diseases, as it can capture subtle changes in eye movements and gaze patterns that may reflect the underlying neural dysfunction associated with these diseases that may not be captured by conventional neurophysiological tests or neuroimaging methods. Eye tracking can also be used to differentiate between different neurological diseases based on their characteristic eye movement abnormalities, and provide objective and quantitative data that can help monitor disease progression and evaluate treatment effects [8, 10-15].

Eye tracking is a promising tool for the early detection and diagnosis of neurological diseases, and has several advantages over conventional diagnostic methods, such as being non-invasive, low-cost, easy to administer, objective, and sensitive. However, the utilization of eye tracking as a diagnostic tool for neurological diseases is still limited by several factors, such as the lack of a standardized eye tracking protocol and metrics for different neurological diseases; the lack of normative data on eye tracking performance in healthy and diseased populations; the lack of validation studies to establish the diagnostic accuracy, sensitivity, specificity, predictive value, and clinical utility of eye tracking for different neurological diseases; and the lack of integration of eye tracking with other diagnostic methods and biomarkers [7, 9, 16].

People with neurological diseases in South Africa often face challenges in getting diagnosis, treatment, and care because of their socioeconomic status. Eye tracking can help to overcome these challenges by offering a diagnostic tool that is more affordable and accessible than other methods. Hence, there is a need for more rigorous and comprehensive studies to establish the validity, reliability, accuracy, and utility of eye tracking study as a diagnostic tool for neurological diseases.

Utilization of Automated Eye Tracking Study as an Ancillary Test in Neurological Disorders

HYPOTHESIS

Eye tracking can be used as an ancillary diagnostic tool for various neurological diseases that present with eye movement abnormalities.

AIMS & OBJECTIVES

The main objective of this research is to investigate the utilization of eye tracking as a diagnostic tool for neurological diseases. The specific objectives are:

- To review the current literature on eye tracking in neurological diseases and identify the key features in this field.
- To develop standardized eye tracking protocol and metrics which could be universally employed for different neurological diseases.
- To collect normative data on eye tracking performance in healthy subjects across different age groups.
- To collect eye tracking data from patients with different neurological diseases.
- To compare eye tracking performance between patients with different neurological diseases and healthy subjects.
- To correlate eye tracking data with other diagnostic tests and biomarkers (such as neuroimaging, blood tests, or cerebrospinal fluid analysis).
- To evaluate the comparative and additive diagnostic value, and clinical utility of eye tracking for different neurological diseases.

The implications of this research are:

- To advance the scientific knowledge and understanding of eye tracking as a diagnostic tool in the diagnosis of neurological diseases.
- To provide recommendations for the utilization of eye tracking in the diagnosis of neurological diseases.
- To propose potential avenues of future research to improve the efficiency, reliability, and accuracy of the diagnosis of neurological diseases using eye tracking.

DESIGN & METHODOLOGY

Setting

Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal, Nelson Mandela School of Medicine, Durban, South Africa

Study Duration

01 Jan 2024– 01 July 2024

Recruitment of Participants

The participants of this research will be selected by Purposive Sampling from the Neurology clinic and ward at Inkosi Albert Luthuli Central Hospital. The study will be carried the same day after their routine clinic visit (no extra visits needed) or while admitted. The participants will be divided into two groups: healthy subjects and patients with neurological diseases.

The potential participants will receive general information about the study by their routine caring doctors. Those interested will be assessed by the investigator for eligibility through a health and diagnostic review, if considered eligible, they will be provided with detailed description of and information about the study. Those individuals meeting criteria and are consented, will be enrolled in the study.

Information will be provided in two languages (English and Zulu), and a nurse volunteer will assist with translation. There will be no financial costs, neither will there be any compensation whether monetary, complementary medical insurance (aid), or otherwise for participants as a result of participating in the study. The principal investigator will conduct the entire research. Should there be any need for co-investigator, it will be requested and amended accordingly.

Inclusion Criteria

- Age 18 years or older
- Normal or corrected-to-normal vision.
- Ability to understand and follow instructions.
- Willing and able to give informed consent.
- Clinically and physically able to complete eye tracking study.

Exclusion Criteria

- Previous neurological or psychiatric disorders
- History of eye diseases or surgeries
- History of head injuries or concussions
- History of illicit substance abuse
- Current use of medications that may affect eye tracking performance.
- Any other condition that may interfere with eye tracking testing

Materials

We will use the Tobii Pro Spectrum device, which is a screen-based eye tracker that captures gaze data at speeds up to 600 Hz through its cameras. It has high precision and allows for natural head movement during recording. We will use the Tobii Pro Lab software to present stimuli and record various eye tracking metrics during the presentation, which is stored as raw data files.

We will develop an analytic software which provides a quantitative and qualitative report. We will develop video stimuli presentations which will be presented on a 24-inch monitor with a resolution of 1920 x 1080 pixels. The participants will sit approximately 60 cm away from the monitor, with their head position adjusted by a chin rest.

Methods

This research will employ a mixed-methods approach that combines quantitative and qualitative methods. The research will consist of four phases:

- **Phase 1: Literature review**

  A systematic review will be conducted to extrapolate quantifiable eye movement parameters with abnormality signatures in neurological diseases.

- **Phase 2: Test protocol development and validation**

  Based on the findings of the literature review, a standardized eye tracking protocol and metrics will be developed for potential diagnosis of different neurological diseases. The protocol will consist of a set of visual stimuli that elicit relevant specific neural processes which can reflect various aspects of brain function and its localization. The protocol and metrics will be confirmed using expert opinions and pilot testing.

  Qualitative measures will consist of graphic path patterns that reflect eye movement characteristics during stimulus presentation through developing an analytic software to perform:

  - Importing raw eye tracking data recorded by eye tracker for further processing
  - Filter noise and outliers
  - Process the raw eye tracking data to analysable data
  - Calculate various eye tracking metrics from processed data
  - Plot descriptive graphs and tables with metrics
  - Provide a complete report in a PDF file format

  Quantitative measures will consist of various eye tracking metrics (descriptive and inferential statistical analysis used is described in statistics section) as follows:

  - Fixation: congruency, accuracy, deviation, and nystagmus/intrusions (right beating, left beating, square-wave jerks, and opsoclonus
  - Saccades: latency/phase, hypermetric, hypometric, accuracy and speed
  - Smooth pursuit: Gain between speed of target and eye tracking, presence of saccadic intrusions, presence of catch-up saccades
  - Specifications: amplitude and velocity
  - Anti-saccades: accuracy, error rate and phase lag

  The stimuli and tasks will include:

  - Five fixation tasks, where the participants will be asked to fixate on a target dot which appears on the center, right, left, up, or down on the screen in horizontal and vertical planes, each for 10 seconds. These tasks will measure the stability of fixation, conjugation, accuracy, deviation, and the presence of involuntary eye movements (e.g., nystagmus, intrusions).
  - Four saccade tasks in two presenting speeds of slow and fast, where the participants will be asked to make quick eye movements between two target dots on the screen. The dots will appear successively on the horizontal plane (right and on the screen), and on the

vertical plane (up and down on the screen) at different speeds (slow and fast). This task will measure the accuracy, continuity, and latency of saccades (e.g., hyper/hypometric saccades).

- Four smooth pursuit tasks in two presenting speeds, where the participants will be asked to follow a moving target dot on the screen with their eyes. The dot will move horizontally or vertically at different speeds (slow and fast). This task will measure the ability to track a moving target smoothly, continuously, and accurately (e.g., catch-up saccades, lag).
- Two anti-saccade task, where the participants will be asked to make quick movement to the opposing target dot on the screen when one flashes. Three dots will appear on the horizontal plane (center, right, and left side of the screen), and on the vertical plane (center, up, and down side of the screen). When either of the side target dots flashes the participant looks to the opposite target dot that is not flashing. This task will measure the volitional inhibition of reflexive saccade.

- **Phase 3: Data collection**

  Normative data on eye tracking performance will be collected from healthy subjects across different age groups (16-40, 41-60, and >60 years) and cultural backgrounds (African, Asian, European). The subjects will undergo eye tracking testing using the standardized protocol developed in phase 2.

  Eye tracking data will also be collected from patients with different neurological diseases. The variety of these diseases is limited by the type of patients who present during the data collection period. The patients will be recruited from neurology clinics or the hospital ward who are diagnosed according to the established criteria for each disease. The patients and healthy subjects will undergo eye tracking testing using the same protocol.

- **Phase 4: Mechanism exploration and correlation assessment**

  A subset of data from patients who have other diagnostic tests in their medical records (such as neuroimaging, neurophysiology, blood tests, or cerebrospinal fluid analysis) will be selected in this phase to compare diagnostic strength and to correlate the eye tracking data with structural and functional brain changes associated with each neurological disease.

Procedure

The participants will undergo an eye tracking test session, where they will be seated or positioned (patients on wheelchair) in front of a desk while resting their chin on a chinrest. The participants will be presented a set of visual stimuli on a screen in front of them located at a specific distance from the participants.

The visual stimuli consist of videos of fixed or moving dots. Each visual stimulus will be presented for a fixed duration (e.g., 10 seconds). The participants will receive instruction for each stimulus.

The eye tracker cameras located bellow the screen will capture the eye movements and gaze patterns of the participants during the stimulus viewing. The eye tracker device will be connected to a computer that will record and store the raw eye tracking data. The eye tracking test session will take about 30 minutes and will be conducted by the investigator.

OUTCOMES

This research is expected to produce the following outcomes:

- A review of the literature on eye tracking in neurological diseases.
- A universal standardized eye tracking protocol and metrics for diagnostic eye tracking test.
- Normative values on eye tracking performance.
- Comparison between eye tracking performance in healthy subjects and patients.
- Correlation between eye tracking performance in patients and other diagnostic methods.
- Diagnostic value of eye tracking performance.

STATISTICAL CONSIDERATIONS

The sample size for normative data with a margin of error of 0.4 degree and a 95% confidence interval, with an assumptive standard deviation for specified metrics in the population is estimated at 8 participants per group for a significance level of 5% and a power of 80%. The total of minimum 24 healthy participants is estimated for normative data calculations. However, this number subject to change based on gathered data's variances. The number of participants with neurological disease is limited by the type of patients who present during the data collection period.

The suitable statistical analysis for assessment and analysis will be conducted and may include:

- Descriptive statistics: To summarize the characteristics and eye tracking performance of the participants using measures of central tendency, dispersion, and distribution.
- Normative methods: To establish the normal ranges and variations of eye tracking performance in healthy populations using methods such as z-scores, percentiles, or confidence intervals.
- Multivariate analysis: To examine the effects of confounding factors (such as age, gender, education, or culture) on eye tracking performance in healthy populations using methods such as analysis of variance (ANOVA), analysis of covariance (ANCOVA), or multiple regression.
- Inferential statistics: To compare eye tracking performance between patients with different neurological diseases and healthy subjects using methods such as t-tests, Mann-Whitney U tests, or Kruskal-Wallis tests.
- ROC analysis: To evaluate the diagnostic accuracy, sensitivity, specificity, predictive value, and clinical utility of eye tracking for different neurological diseases using methods such as receiver operating characteristic (ROC) curves, area under the curve (AUC), or cut-off points.
- Correlation analysis: To explore the relationship between eye tracking data and neuroimaging data, other diagnostic methods, and biomarkers for neurological diseases using methods such as Pearson's correlation coefficient, Spearman's rank correlation coefficient, or partial correlation coefficient.

SAFETY & RISKS

Eye tracking is generally considered a safe and non-invasive technique, as it does not involve any physical contact, radiation, or stimulation of the eyes or the brain. Some participants may experience anxiety during the eye tracking testing, especially if they feel that their eye movements are being judged. Therefore, the

comfort, privacy, and well-being of the participants during and after the eye tracking testing will be ensured by providing adequate instructions, breaks, feedback, and support.

ETHICAL CONSIDERATIONS

This study will be conducted in full accordance with all applicable local and regional research policies, procedures, laws, and regulations, and the World Medical Association Declaration of Helsinki on Ethical Principles for Medical Research Involving Human Subjects. Necessary approval from the relevant authorities, committees, review boards, or institutions where this study is conducted at, will be obtained.

All episodes of noncompliance will be documented. Data collection, storage, access, reporting, and protection will be anonymized and secured in compliance with the HIPAA regulations and standards for data security, privacy, and breach notification.

The purpose, procedures, risks, and benefits of the research and its voluntary nature in an utterly voluntary, comfortable, and informed manner, will be explained to the participants. All information, guidance, and consent will be provided in English and Zulu languages. Only patients who are able to provide informed consent will be eligible for participation.

There will be no financial costs or compensation. There will be no vulnerable participants in the study. There will be no compromise in continuity of care. No extra visits, interventions, other diagnostic tests, or services will be conducted for the sake of this study. All required data is deducted from the patient medical records based on routine diagnostics done for patients.

Data Safety and Monitoring Board (DSMB)

- Prof VB Patel
  Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal
  Patelv@ukzn.ac.za
- Dr Ferzana Amod
  Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal
  Amodf@ukzn.ac.za
- Dr Kaminie Moodley
  Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal
  Moodleyk4@ukzn.ac.za
- Dr Hlubi Dlwati
  Department of Neurology, Inkosi Albert Luthuli Central Hospital, University of KwaZulu-Natal
  dlwathim@ukzn.ac.za

REFERENCES

1. Organization, W.H., *Mental health: neurological disorders*. WHO, 2016.
2. Thakur, K.T., et al., *Neurological Disorders*, in *Mental, Neurological, and Substance Use Disorders: Disease Control Priorities, Third Edition (Volume 4)*, V. Patel, et al., Editors. 2016, The International Bank for Reconstruction and Development / The World Bank © 2016 International Bank for Reconstruction and Development / The World Bank.: Washington (DC).
3. Benbadis SR, B.S., Cavazos JE, et al., *Drugs and Diseases Articles on Neurology*. Medscape Reference, 2021.
4. Siuly, S. and Y. Zhang, *Medical Big Data: Neurological Diseases Diagnosis Through Medical Data Analysis*. Data Science and Engineering, 2016. **1**(2): p. 54-64.
5. Tawa, N., A. Rhoda, and I. Diener, *Accuracy of clinical neurological examination in diagnosing lumbo-sacral radiculopathy: a systematic literature review*. BMC Musculoskeletal Disorders, 2017. **18**(1): p. 93.
6. Brunyé, T.T., et al., *A review of eye tracking for understanding and improving diagnostic interpretation*. Cognitive Research: Principles and Implications, 2019. **4**(1): p. 7.
7. Schalén, L., N.G. Henriksson, and I. Pyykkö, *Quantification of tracking eye movements in patients with neurological disorders*. Acta Otolaryngol, 1982. **93**(5-6): p. 387-95.
8. Terao, Y., H. Fukuda, and O. Hikosaka, *What do eye movements tell us about patients with neurological disorders? - An introduction to saccade recording in the clinical setting*. Proc Jpn Acad Ser B Phys Biol Sci, 2017. **93**(10): p. 772-801.
9. Umeda, Y., *The eye-tracking test*. Ann Otol Rhinol Laryngol Suppl, 1980. **89**(4 Pt 3): p. 1-18.
10. Abul Hassan, M., et al., *Approach to Quantify Eye Movements to Augment Stroke Diagnosis With a Non-Calibrated Eye-Tracker*. IEEE Trans Biomed Eng, 2023. **70**(6): p. 1750-1757.
11. Avendaño-Valencia, L.D., et al., *Video-based eye tracking performance for computer-assisted diagnostic support of diabetic neuropathy*. Artif Intell Med, 2021. **114**: p. 102050.
12. Beltrán, J., et al., *Computational Techniques for Eye Movements Analysis towards Supporting Early Diagnosis of Alzheimer's Disease: A Review*. Comput Math Methods Med, 2018. **2018**: p. 2676409.
13. Bueno, A.P.A., J.R. Sato, and M. Hornberger, *Eye tracking - The overlooked method to measure cognition in neurodegeneration?* Neuropsychologia, 2019. **133**: p. 107191.
14. Nguyen, M.N.L., et al., *Tracking Eye Movements for Diagnosis in Myasthenia Gravis: A Comprehensive Review*. J Neuroophthalmol, 2022. **42**(4): p. 428-441.
15. Samadani, U., et al., *Eye tracking detects disconjugate eye movements associated with structural traumatic brain injury and concussion*. J Neurotrauma, 2015. **32**(8): p. 548-56.
16. Tao, L., et al., *Eye tracking metrics to screen and assess cognitive impairment in patients with neurological disorders*. Neurological Sciences, 2020. **41**(7): p. 1697-1704.