Modeling Cognitive Process from Visual Attention Geometry in AR/VR:

A Machine Learning Approach

A thesis submitted by

Ioana Lupascu

in partial fulfillment of the requirements for the degree of

Master of Science

in

Human Factors Engineering

Tufts University

February 2024

Advisor: Daniel Hannon, PhD

Committee:

James Intriligator, PhD

Farakh Zaman, PhD, Lt Col, USAF

Abstract

The ecological theory of perception maintains that visual optic arrays contain geometric patterns that afford information to a perceiver, influencing interactions with one's environment. In this thesis, the author develops methods of visual field decomposition in 2D optical arrays and 3D spaces to characterize visual attention patterns and cognitive states from data previously collected in a Virtual Reality (VR) simulator. A novel algorithm was developed to isolate markers of human visual attention dynamically as gaze vectors adapt continuously with movement through space.

Supervised and unsupervised machine learning models were constructed and trained on a combination of visual attention features to understand the contributions of distinct markers of attention to psychophysical behaviors. Hypothesis-driven studies and respective results aided the selection and validation of meaningful classification categories and boundaries in the areas of: attention shift, response time, and decision-making. This thesis yielded machine learning models that successfully classified cognitive states from patterns in 2D and 3D visual attention. Predictive analytics and visualization methods developed in this work show the potential for AI-enhanced eye tracking technology to augment decision-making during visual search and navigation applications.

Acknowledgments

With utmost gratitude, thank you to Dr. Dan Hannon, Dr. James Intriligator, and Dr. Farakh Zaman, Lt Col, USAF for allowing me the opportunity to join this research program and to expand upon the knowledge and foundation that you have established. Collaborating with all of you has been an honor and an opportunity of academic rigor and depth beyond my expectations.

Dr. Hannon, thank you for trusting me to take this next step in the project, for introducing me to the space of ecological visual perception, and for advising this thesis. The study designs in this work are shaped by your teachings on ecological attention patterns and inspired by your work on optical flow. Your constant reminder to uphold a sound data process and clear experimental design was a compass that helped me navigate the often nebulous world of machine learning.

Dr. Zaman, thank you for your consistent support throughout my time completing graduate studies at Tufts during the pandemic. The project that you built and the studies you completed provided a rich platform upon which I was able to design and test the methods I present in this thesis. Your support and encouragement gave a sense of purpose to many of us in the graduate program in an otherwise very dark time in this world, and I sincerely hope that the work I have done here continues to support you in your mission.

Dr. Intriligator, your guidance on this project and during my academic years at Tufts, and the clarity and enthusiasm that you brought to every research conversation and to your courses were a steady source of inspiration and possibility. Thank you for welcoming me into your lab and on this project, and for your passion for pushing the boundaries of what is possible in multisensory design and AR/VR. Thank you to Tufts University Mechanical Engineering and Data Science departments, for the knowledge, data, and tools that made this project possible.

Thank you to my mother and my father, whose paths as engineers paved the way for mine, and to my grandmother and grandfather, whose wisdom continue to guide me from above.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 01:

INTRODUCTION

1 INTRODUCTION

1.1 Attention

The fundamental essence of human attention is universally understood, but specific dynamics of attention allocation remain elusive. Conceptual understanding of attention is derived from several schools of thought. From an Eastern philosophy perspective, attention includes passive cognitive elements such as mindfulness, watchfulness, and allowing one's consciousness to be fully available to the present moment. From a Western cognitive approach, attention is imbued with a more dynamic ability to direct focus. Selective attention, then, is concerned with the mechanism of placing focus on a given object, location, or stimulus. The cognitive resources available in the selective allocation of attention towards a task are limited. Carrasco (2014) reviews an extensive group of studies which explain the concept of covert attention as an indicator of "limited cognitive and brain resources" (Carrasco, 2014). The amount of resources necessary to carry out a cognitive task are captured by the concept of cognitive load. While our understanding of the process of cognitive load allocation has grown substantially in the past 25 years (Carrasco, 2014), gaps remain in our understanding of the signal pathways involved in allocating visual attention.

1.2 Ecological Perception

James Gibson's theory of ecological perception challenged traditional views of visual perception by suggesting that information from the 3-dimentional world is presented onto the retina in a 2-dimentional surface. Instead of complex processing and interpretation of rich 3D features into space, Gibson proposed a mechanism of dimensionality reduction, by which 3D information is collapsed onto an optic array (Gibson, 1977). The 2D optic array contains geometric patterns which, according to Gibson, *afford* information about the environment to the perceiver. This information, or *affordances*, can be relied upon by the perceiver to make decisions about interacting with one's environment. In Figure 1-1 below, for example, optical angles remain constant and afford the perceiver the possibility to deduce the location of the horizon and afford the perceiver the necessary information to direct their motion forward.

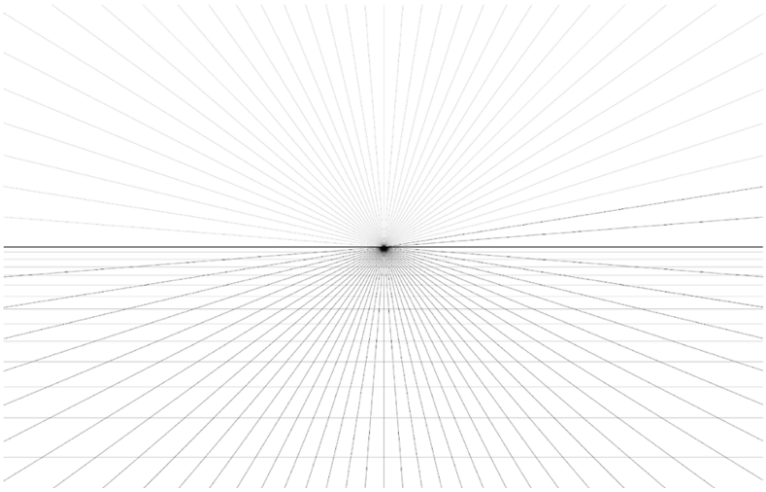


Figure 1-1. Patterns in a 2D Optic array (Ecological View of Perception)

Note. From *Room perspective grid background 3d Vector illustration architecture model projection.* [Images], by v_ctoria. 2024 AdobeStock.

In their work on self-motion and optical flow, Warren and Hannon were among the first to explore computational methods for *visual decomposition* of the field of view (Warren & Hannon, 1988). They used measures of *visual angle* to calculate difference vectors between neighboring elements relative to a destination point. This thesis work leverages the concept of geometric patterns between objects in visual attention to study and model cognitive process.

1.3 Purpose

In this study, statistical methods of correlation and classification are applied to study the relationship between optical gaze patterns and cognition**.** Data from experiments conducted in a subterranean military simulator are examined for correlation between variables under a hypothesis-driven research phase, then applied to train new hypothesis-generating machine learning models. This thesis aims to gain insights into the process of *visual attention allocation* under psychological stress.

The author develops methods of *visual field decomposition* in 2-dimentional and 3-dimentional spaces to characterize gaze patterns and cognitive states from eye-tracking data collected in a Virtual Reality (VR) simulator.

This work has three goals:

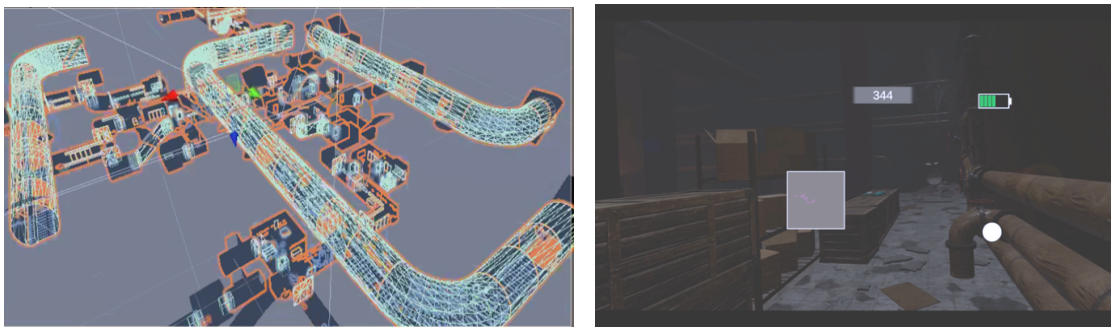
1. First, a series of analyses are conducted to understand and characterize patterns of visual attention from the geometry of visual attention.
2. Second, known theoretical frameworks of attention are applied to the VR space to generate new insights towards the development of navigational systems and training systems equipped with predictive decision support.
3. Third, this work evaluates the potential of Machine Learning predictive analytics to accurately predict cognitive activity from visual geometry in 2-dimensional and 3-dimensional Space.

Methods of studying visual attention allocation have been developed and applied in several key industries, including the study of driving behavior in the automotive space, human-in-the-loop aerospace technology (Glaholt, 2014), and autonomous control of vehicles with the advent of artificial intelligence (Fridman et al., 2018). This work expands on previous work in applied cognitive psychology and vision research to contribute new algorithmic methods of studying the mechanisms of visual attention allocation under cognitive load in a VR simulator. The complete research philosophy is detailed in Chapter 2.2.

1.4 Original VR Simulator Eye Data

The original VR simulator was developed as an alternative training environment for combat troops preparing to enter environments that are otherwise dark, tight, and physically challenging to navigate (Zaman et al., 2021). The original experimental design and simulator design and construction were conducted and led by Zaman, Intriligator, Hannon et. al at Tufts University as part of a five-year study analyzing Dynamic Information Needs Analysis in subterranean spaces (*United States Army Grant # A452001 AR9007*). The VR simulator allows researchers to study human behavior in subterranean spaces and similar environments where users have limited access to information and lack satellite access that would enable GPS systems for navigation.

The following images depict the original VR tunnel setup in Unity (Zaman et al., 2021). Figure 1-2 shows one example layout. This layout depicts (a) the 3D environment walked by participants during a run, and (b) the screen-level first-person participant view into the 3D VR space with the 2D Heads Up Display.



(a) 3D view, full depth (b) 2D screen-view

Figure 1-2. Unity simulator (a) full environment and (b) first-person view (Zaman, 2022)

One **run** is defined as **one subject** "walking" virtually through **one layout** (of four) with various conditions in the VR environment to complete a series of tasks accompanied by distractor tasks. Biometric data such as heart rate, pupil size, blink rate, location in the 3D environment and gaze location were recorded for all trials.

1.5 Studying Attention in 2-Dimensional vs 3-Dimensional Space

In this thesis work, exploratory modeling is conducted to characterize visual attention patterns from user eye-tracking data collected in the previously described VR simulator.

This work examines visual attention patterns in the 2D optical array and in 3D space. The 3D visual decomposition method is complex and is described in detail in later chapters as it builds upon the foundation laid through the 2D method. First, a series of analyses were performed in the 2D-screen plane to establish the extent to which correlations exist between screen-plane visual attention parameters and cognitive markers. For this, a baseline relationship is first established between select 2D geometrical features of attention and key attributes of the participant's responses to tasks. The 2D analysis relies on the Tobii-defined 2D-screen coordinate system. Experimental designs of studies performed in 2D space are defined in Chapter 4.

Visual attention parameters and calculations in 3D space are explained in Chapter 5. For the 3D analyses, a coordinate system is dynamically defined at the user's location and continuously updated as the participant moves through space.

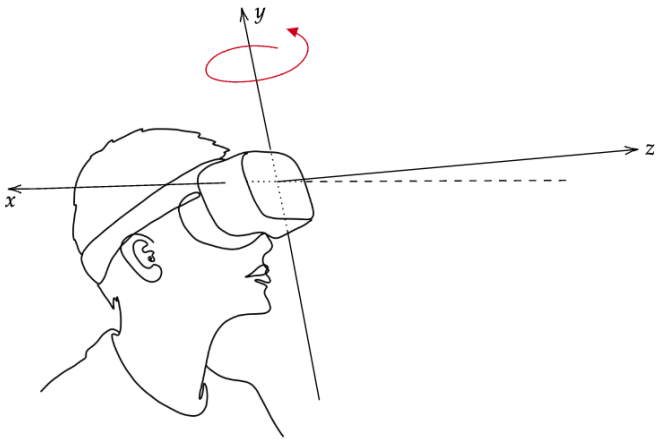


Figure 1-3. Representation of VR Coordinate System in 3D

Note: Coordinate system and direction of rotation adapted to image from: *Man in Future Living Metaverse* [Vectors], by Cinz, PNGTree.https://pngtree.com/freepng/man-in-future-living-metaverse_6960080.html

Experimental deigns of studies performed in 3D space are defined in Chapter 3, and all calculations and trigonometric transformations for the 3D-geometry analyses are detailed in Chapter 5.

1.6 Research Questions

In this Thesis work, studies are conducted in (4) Research Areas:

1. Relationships between Cognitive Load Latency and 2D Perception Markers
2. Differentiating Between Presence and Absence of Stimulus
3. Attentional Shift Behaviors towards Stimulus
4. Predicting Decision-Making

The following questions are used to drive inquiry into cognitive state and gaze pattern. The research questions proposed here are answered in a 2-dimensional spatial domain, a space-time (temporal) domain, and in a 3-dimensional spatial domain.

1. To what extent can **cognitive load** be understood through markers of gaze behavior?
    a. To what extent is there a correlation between **gaze dispersion** and cognitive load?
    b. To what extent is there a correlation between **visual angles**, **gaze path**, and **head rotation speed** and cognitive load latency?
    c. What proportion of the variance in cognitive load latency is attributable to gaze movement and head rotation at the time of signal discernment?
    d. What proportion of cognitive load latency is attributable to gaze path movement towards the **positioning of the cognitive distraction stimulus**?

2. Can **machine learning models** be trained to detect psychological markers of cognitive load from visual allocation patterns and other biometrics?
    a. To what extent can variability in visual attention allocation predict cognitive load?
    b. To what extent can a decision point be predicted from gaze behavior?

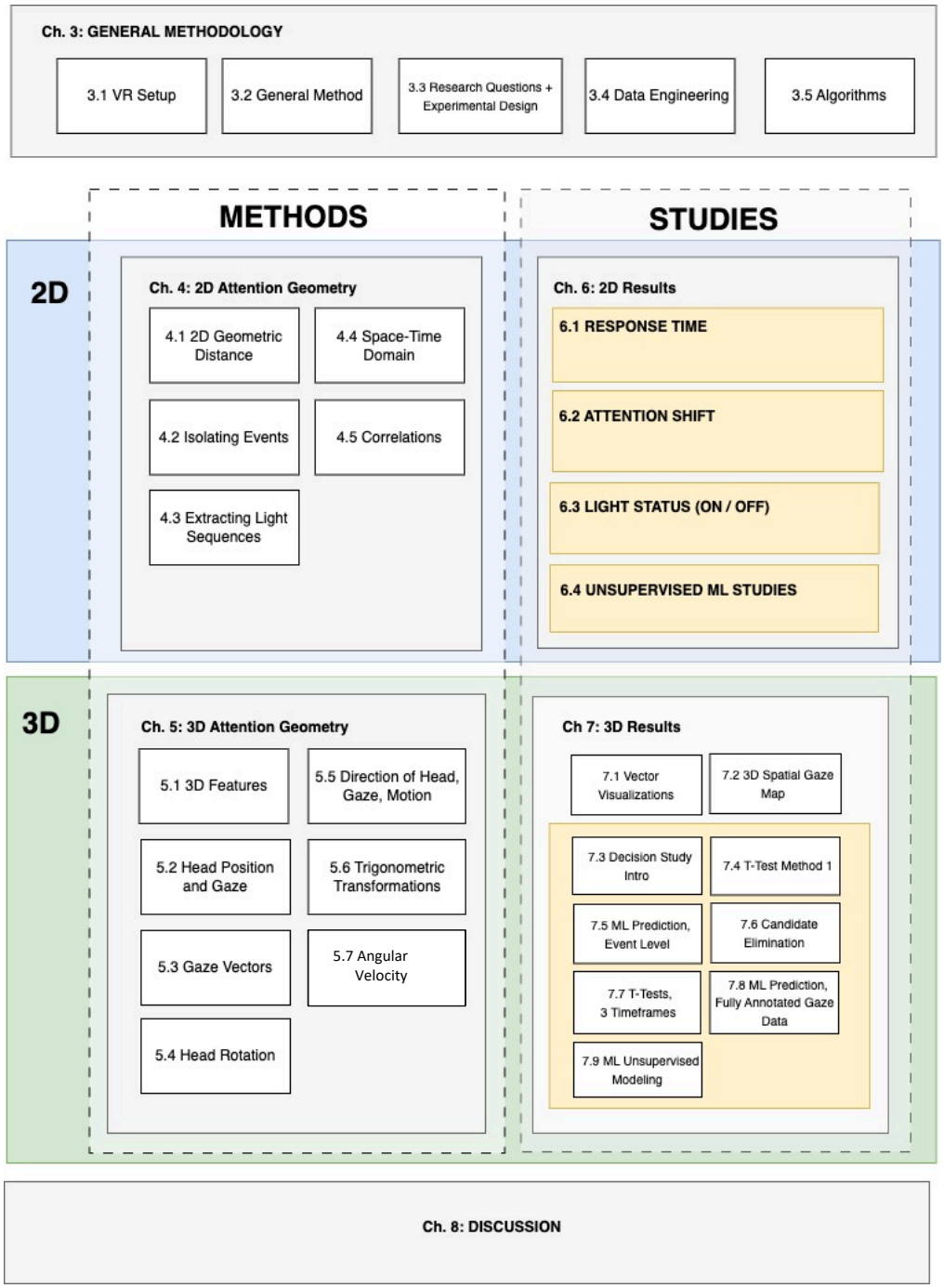
Several methodological questions are also explored throughout this work:

3. To what extent does Virtual Reality eye tracking model real world visual attention allocation?
4. What useful gaze metrics can be derived from eye tracking data in a Virtual Reality environment?
5. What are effective techniques for analyzing VR/AR gaze data in 2D and 3D spatial domains and space-time temporal domain?
6. What are appropriate Supervised and Unsupervised Machine Learning models that can address each research question?

1.7 Thesis Outline

The schematic in Figure 1-4 guides the reader through the upcoming methodology, results, and discussion chapters. Because the methods developed, and the studies conducted, span both 2D and 3D geometry, it is useful to refer back to this map which shows how each study parameter was derived in the respective space.

Figure 1-4. Thesis Outline Diagram



CHAPTER 02:

THEORETICAL FRAMEWORK

2 THEORETICAL FRAMEWORK

This chapter details the theoretical framework that guides this thesis work. Literature is reviewed to frame the context for the visual attention studies in 2D and 3D. Known quantitative tools of working with metrics of visual attention are explored. Machine Learning concepts that guide the analyses in later chapters are introduced.

Chapter 2 Topics:

- 1. Eye Movements
- 2. Visual Search
- 3. Discerning a stimulus
- 4. How to quantify visual attention
  - visual decomposition
  - gaze dispersion
  - patterns over time
- 5. Eye Tracking to Understand Cognitive mechanisms
- 6. Machine Learning Modeling
    - Supervised Learning
        - Linear Regression
        - Logistic Regression
        - Decision Trees
        - K Nearest Neighbors
    - Unsupervised Learning
        - k-Means
        - Principal Component Analysis

2.1 Eye movements

Purves et al. define four distinct types of eye movements: "saccades, smooth pursuit movements, vergence movements, and vestibulo-ocular movements." (Purves et al, 2001). Tobii, the manufacturer of the eye tracking software and SDK used in this thesis work, define several additional movements with more granularity: micro saccades, tremors, and drifts. (Miseviciute, 2023). Fixation refers to the behavior by which the eye maintains the focus steadily for some time on one location. The eye movements have both independent and interconnected functions and are driven by different ocular mechanisms. Information in the field of view is processed in various ways in each visual behavior. In his 2017 Google TechTalk, Dixon Cleveland introduces the mechanism and function of *fixation*, observing that ocular muscles "hold the eye still for the 250ms it takes to get enough photons to get a good image of the foveal area" (Cleveland, 2017). *Fixation eccentricity, then,* refers to the position of the fovea relative to the point of fixation.

A *saccade* is the fast motion of the eye from one locus of fixation to another, at approximately four times per second (Cleveland, 2018). During a saccade, "processing information from the visual field is suspended" (Glaholt, 2014). This phenomenon, known as *saccadic suppression,* was identified as early as 1898 by psychologists Erdmann and Dodge, and the mechanism of suppression has been studied throughout the last century (Matin, 1974). Santini, Rucci, et al point out that the function of fixation has long been debated, with some arguing that that the sole function of stabilizing the eye gaze onto one location is to keep the image from fading, and others exploring additional functionality in fixation (Rucci et al, 2007).

2.1.1 Visual Search

Several theories are used to explain the mechanisms by which we search our environment for features. *Feature integration theory* is one notable concept, in which the search through visual elements includes the main target and the background. Two kinds of searches have been

previously defined. In *feature search*, only one feature is required to detect the target, and features are easy to distinguish from the background array, while in *conjunction search*, two features must be used to identify a target (Treisman & Gelade, 1980). *Inattentional blindness* refers to the phenomenon by which paying attention to one aspect of a visual scene makes us blind to other aspects in the scene, so a signal can be missed out of distracted attention. One famous example of this phenomenon is Simons and Chabris's gorilla experiment, in which observers failed to notice a person dressed in a gorilla suit in the middle of a basketball game because their attention was focused on other features of the scene (Carpenter, 2001).

2.1.2 Discerning a Stimulus

Carrasco refers to attention as the “mechanism that turns looking into seeing” (Carrasco, 2014). Visual attention refers to the attention we allocate to objects in our visual field. In the neuropsychology of visual attention, distinct pathways have been identified for spatial orienting (sense of direction and location in space) and object orienting (focus on what objects are). As the human mind discerns signal from background, it reaches one decisive moment in which the identity of a stimulus is noticed. In their work, "Attention and Consciousness", De Brigard and Prinz identify the mechanism behind directing attention towards a signal as "involving an increase in interneurons, which send inhibitory signals to pyramidal cells that encode sensory information" (De Brigard & Prinz, 2010). This is followed by a process by which the stimulus is encoded to working memory, leading to an ability to not just perceive but actively identify an object. Once perceptual pathways are activated, sensitivity increases, causing an increase in contrast and greater ability to discern signal from background. This increase in capacity to identify a salient object happens simultaneously at "both the neural and phenomenological level" (De Brigard & Prinz, 2010). At the moment of noticing, the stimulus triggers a response in the centers associated with executive control of attention.

2.1.3 Quantifying Visual Attention

Various quantitative methods of psychophysiological research have been applied to the study of attention. In the space of driver attention, functional methods of assessing visual attention allocation include eye gaze concentration, situation awareness, and hazard perception. (Louw,

2017). Louw et. al assessed drivers' attention by using *gaze dispersion* to measure visual attention allocation towards different areas of interest in the visual field (Louw & Merat, 2017). In this thesis work, the author includes Louw's method of gaze dispersion modeling as one of several metrics to study visual attention allocation in the VR environment.

2.1.4 Visual Patterns over Time

Traditional statistical modeling is useful in establishing relationships between phenomena. For example, the response time to a stimulus can be compared between separate groups of people and for different conditions. Human behavior can be categorized by meaningful markers and therefore compared between and within groups. In gaze research, a simple statistical inquiry can allow a researcher to compare the differences in gaze fixation or gaze dispersion for different subjects at any point in time. However, simple statistical inquiry is limited in its ability to infer cognitive state from eye tracking data as these analyses rely on averages of data over time. Important nuances in thought and behavior that unfold over time are lost when the time dimension is reduced for statistical comparison. This thesis work seeks to understand gaze pattern and mechanisms in space and time domains.

2.1.5 Application

In the field of human computer interaction, tracking eye movement has both external and internal applications. When applied to entirely external inquiry, eye tracking is concerned with the *direction of gaze* and the *object of interaction*, such as areas of interest on a computer screen in consumer and marketing research. Internal applications study eye movements in order to make inferences about cognitive process. One common application of interest for eye tracking technology is the study of cognitive workload and fatigue in drivers (Fridman et al., 2018). Eye tracking technology tends to be less expensive and more accessible than other methods of cognitive state measurement such as electroencephalography, or EEG.

Vortmann et al (2021) explore the distinction between internal and external applications of gaze observation. The use of eye tracking in visual pattern discernment and inference of cognitive state has strong applicability in the space of real-time adaptive displays and integration with head-mounted displays such as the HoloLens smart glasses (Vortmann et al., 2021). However, while useful to the study of external phenomena, when pointed inward, eye tracking technology is limited in its ability to infer and classify mechanisms of perception.

At any point in time, the mind takes in inputs from various stimuli across all senses. In his Google TechTalk, Cleveland describes the consequences of multisensory processing in attentional shift (Cleveland, 2017). For visual attention to be shifted, and for a person to saccade their gaze from one fixation point to the next, a decision is made at the time when the brain takes in several stimuli and decides on the direction of the eye's movement. Each input signal carries a potential to draw the attention towards it. Auditory, visual, haptic, and emotional factors could all be contributing to available information. In the end, only one of these inputs crosses a threshold, visual attention is selectively placed on the new locus of fixation, and the eye saccades (Cleveland, 2017). Cleveland points out that the specific mechanism of attention allocation remains unknown, and a worthwhile area of study in the space of visual attention research. Modern eye tracking technology cannot see probabilities of specific signals to cross the threshold of detection at a moment of decision. Today, we still do not have a method of assessing the mechanism by which signals cross this activation threshold to direct attention allocation (Cleveland, 2017). The thesis examines visual attention patterns in 2D and 3D space to answer this question and understand the factors that contribute to shifts in attention allocation.

2.2 Research Philosophy

To guide the investigation into the defined research questions, two types of inquiry are conducted in this thesis project: (1) a hypothesis-driven research phase asks targeted questions, and (2) an exploratory ethnographic research phase uses machine learning to facilitate unbiased open inquiry into unknown relationships between features.

2.2.1 Hypothesis-Driven Research

For the *hypothesis-driven research phase*, clear testable null and alternative hypotheses are defined for each category of study and tested using statistical techniques to understand relationships between variables. For each question, the null hypothesis is defined to test the absence of relationship for each variable group. One or multiple alternative hypotheses are then proposed to drive inquiry into the expected relationships. Independent Variable(s) are defined and the correlation between of these and corresponding Dependent Variables, or Target Features, are examined.

2.2.2 Hypothesis-Generating Machine Learning Inquiry

In the *machine learning analyses* conducted in this thesis, a broader concept of *hypothesis space* is considered. One concept for *hypotheses* in the context of training machine learning models on multiple input features comes from the seminal work of Russel and Norvig: *Artificial Intelligence: A Modern Approach (2010).* Russel and Norvig introduce the *hypothesis space, H,* as the full set of functions that may best fit the input points to the data. The process of generating a hypothesis in machine learning modeling, therefore, is the process by which the hypothesis space is explored to isolate the function that is most "consistent with the data" (Russel & Norvig, 2010, p.696). Along with the unbounded exploration of the full hypothesis space, an overarching binary hypothesis is presented in each machine learning analysis which tests whether or not a machine learning model can be trained to accurately predict certain phenomena.

Therefore, in the hypothesis-generating research phase, traditional methods of statistical analysis are coupled to machine-learning modeling to generate new hypotheses for study in areas of potential significance. Through supervised modeling, regression and clustering methods are used to understand the contributions of input features towards the distinction between of outcome classes. Meaningful classes are defined for each target variable, and models are trained to discern between distinct categories. The predictive capacity of each machine learning algorithm trained on this data is evaluated against metrics appropriate for each supervised model. For the unbiased exploratory research phase, unsupervised machine learning models are set up to seek hidden relationships between variables. Hypotheses are not defined, instead several machine learning algorithms are deployed to identify clusters of data and relationships that may be unexpected.

2.2.3 Machine Learning Algorithms

In this thesis work, traditional statistical modeling and machine learning predictions are used to find relationships, clusters, and trends of interest from geometrical patterns of visual attention identified in VR eye tracking data. In recent years, much attention has been given to the development and application of time-sensitive statistical modeling in the space of cognitive science. Extracting sequential information from visual patterns and behavior patterns can give us a deeper understanding of human cognitive states. Vortmann et al compare multiple classification algorithms with one dimensional feature sets and deep models used by various research groups to classify attention states based on eye tracking time series (Vortmann et al., 2014). Fridman et al infer cognitive states from eye tracking movements captured using in-car video recordings (Fridman, 2018). Hayashi et al introduce the Hidden Markov Model as a possible tool to infer the cognitive process of pilots based on their eye movements (Hayashi, 2014). In the thesis work presented here, both supervised and unsupervised machine learning modeling techniques are used to gain insights into the data.

2.3 Supervised Learning

Supervised learning models are applied in this thesis work to identify trends through regression and to distinguish between meaningful groups through classification. In classification, data is labelled, and classes are assigned to selected target variables. Then, models are trained to recognize classes based on historical data. In regression, the relationship between variables is examined to understand the extent of the variability in one feature due to one or multiple independent inputs (Russell & Norvig, 2009). This thesis work applies linear regressions, logistic regression, decision trees, k-nearest neighbors (KNN) to model relationships in the space of visual attention allocation where target variables can be defined.

2.3.1 Logistic Regression

Logistic regression is a statistical method which seeks to predict the probability of a dependent variable occurring in relationship to one or multiple independent variables. Three main types of logistic regression support categorizing a dependent target variable into different numbers of classes. The binary logistic regression allows classification into two distinct categories, the multinomial logistic regression supports classification into more than two classes, and the ordinal logistic regression allows for classification of multiple classes in a predetermined order (Russel & Norvig, 2009). Russel and Norvig, in Chapter 18 of their textbook, *Artificial Intelligence: A Modern Approach* explain that, unlike linear regression, which is a continuous function, the logistic regression models the logarithmic odds of the probability of an event occurring (Russel & Norvig, 2009).The logistic regression utilizes the logistic function to map the inputs x, to the output y:

$$y = \frac{1}{(1 + e^{-x})}$$

Stoltzfus explains that the logistic function is visualized as an S-shaped (sigmoid) curve which rises at a calculated rate and maps real-valued numbers into a value between 0 and 1 (Stoltzfus, 2011). This allows for the use of regression techniques for classification, where the logarithmic odds, or log-odds indicate the probability of the input values to map to each defined class:

$$z = log\left(\frac{y}{1-y}\right) = log\left(\frac{P\{Y=1\}}{P\{Y=0\}}\right)$$

The log-odds are then represented as the weights *(w)* learned by the model for each input variable *(x)* and the related bias (*b*).

$$z = b + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$$

In this thesis, logistic regression is used as a machine learning model to understand relationships between features of visual attention that define meaningful binary classes.

2.3.2 Decision Trees

The decision tree is a commonly used supervised classification model which has great value in its simplicity and interpretability. As with the logistic regression, the variable of interest is set as the *target variable* which the algorithm seeks to classify. Algorithm parameters are tuned to achieve an optimal pathway for classification at each decision point. From Russel and Norvig *Chapter 18: Learning from Examples*, the decision tree is defined as representing "a function that takes as input a vector of attribute values and returns a 'decision' - a single output value" (Russel & Norvig, 2020). The decision tree performs a sequence of splits to the data to arrive at nodes (decision points) and leaves, or the final outcome of the prediction.

2.4 Limitations

In order to reduce the problem space to train machine learning models on the combined data from all experimental layouts (Zaman, 2020), this thesis work assumes consistency between runs. This work does not explore the differences in the original four conditions from the Sub-T study conducted by Zaman et al., where the heads-up display shows navigation information to the participants in the following conditions: (1) Navigation Always On, (2) Navigation On Demand, (3) Automatic Navigation at Decision Points (4) Automatic Navigation at Random Points (Zaman, 2021). For the purpose of this study into gaze behavior and cognitive load, the problem space is reduced to analyze the correlation between metrics of visual attention and cognitive load latency uniformly without regard to differences between layouts. For 2D studies which relied on light-task performance, the same process was applied across all layouts and navigational display methods. For the 3D-studies which utilized Navigational Tools requests to isolate behavioral events, Always-On conditions were removed, and all other conditions were included without further distinction in the extracted events.

CHAPTER 03:
METHODOLOGY

3 METHODOLOGY

- 1. Simulator Setup
- 2. 2D General methodology
- 3. 3D general methodology
- 4. Research Strategy
    - Ethnography (no control group)
    - Exploratory Research
    - Hypothesis-Driven
    - Machine Learning
- 5. Time: one point in time (cross-sectional) vs. Longitudinal (multiple points in time)
- 6. Sampling Strategy
    - subject selection
    - equipment limitations
    - Selecting event samples from data
    - All-data with annotation
- 7. Data Engineering
- 8. Data Science

3.1 Simulator Setup

Study participants performed a series of primary and secondary tasks inside four VR subterranean layouts (Zaman, 2020). Subjects were physically seated for the length of the study and "walked" virtually by propelling themselves inside the VR simulation using a handheld controller (Zaman, 2020). The equipment used was the HTC Vive Pro Eye with Tobii eye tracking software. Table 1 shows the hardware, software, and statistical packages that were used for data acquisition, data processing, and AI/ML predictive analytics in this thesis.

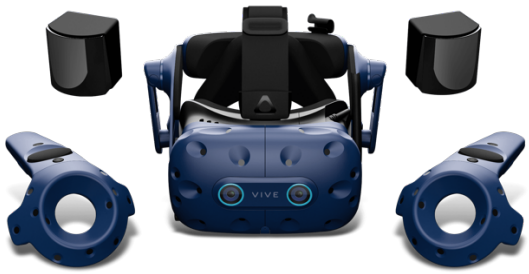


Figure 3-1. *HTC Vive VR Headset and Controller Hardware*
Product Image from: HTC Vive Enterprise*, 2022.* https://enterprise.vive.com/us/product/vive-pro-eye-office/

Table 1. Hardware, Software, Statistical Packages

Hardware	
VR Headset	HTC Vive Pro Eye
Heart Rate Monitor	Polar
PC Analytics and Video	M1 Processor

Software: AR/VR Data Acquisition	
Eye-Tracking API	Tobii Version
DINA Simulated Tunnel	Unity Version 2019.5
Combined Resolution	2880 x 1600 pixels

Software: Statistical Packages	
Anaconda Navigator	Version 2.1.0
Jupyter Notebook	Version 6.5.0, Python v. 3
Spyder Python Dev. Environment	Version 4.2.5, Python v. 3
MATLAB	2016a student version
Microsoft Excel for Mac	Version 16.57

All participants in the original study by Zaman et al were presented with four visual information cues in the static AR Heads Up Display (Zaman, 2022).These included a battery indicating the longevity of the AR display, a GPS direction indicator, an INS map, and a light task. The ***light task stimulus*** is visible in the bottom left side of Figure 3-2a. The system presented users with a light (a) which they had to deactivate (b) by pressing a button. The primary task of the user in the original study was to locate five boxes in the tunnel. The **secondary task** was to turn off the light task when it appeared on screen (Zaman, 2022).

(a) Light ON: task presented to user (b) Light OFF: task deactivated

Figure 3-2. Visual Light stimulus in AR Heads Up Display (a) ON (b) OFF (Zaman, 2021)

This thesis focuses on the participants' interaction with the secondary-task light, or the distraction stimulus. Along with multiple other instructions received before initiating the trial, 48 participants were actively instructed to turn off a distraction light when it appeared (Zaman et al., 2021). This task is referred to as the cognitive load task, and the light stimulus is referred to as the cognitive load light. *Response time latency* is used as a measure of cognitive load (Nilsson, 2017). Response time latency in this work is measured as the total time between the light appearing on-screen to the time the user presses a trigger to deactivate the light, extinguishing the light from the screen. This metric includes the time it took the participant to notice the stimulus and the subsequent reaction time to deactivate the light. Gaze events in this work are classified as "Light ON" if they occurred during the time when distraction light was on, and "Light OFF" is they occurred in the absence of the distraction task.

3.2 General Methodology

3.2.1 2D General Methodology

First, the visual distance between gaze points in the 2D-plane and the distraction light was isolated as an ecological marker of perception in order to study cognitive phenomena such as response time latency and attentional shift towards a stimulus.

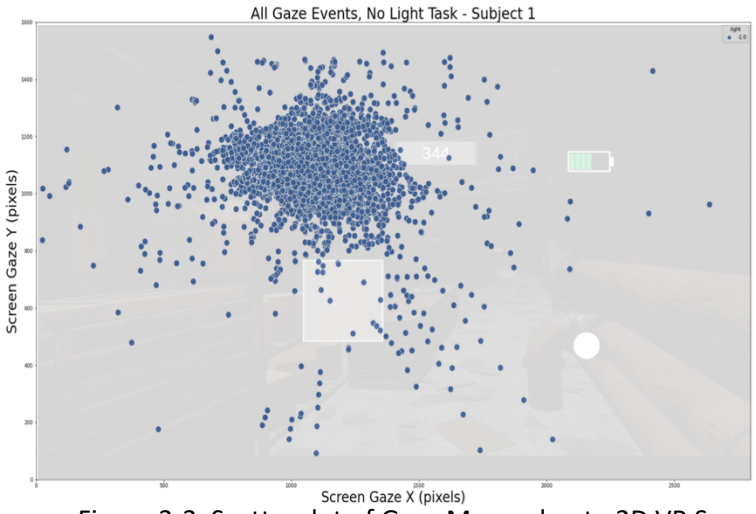


Figure 3-3. Scatterplot of Gaze Mapped onto 2D VR Screen

In order to establish a visual baseline, the distance was calculated from the center of the fixation cluster to the center of the Light Stimulus. Figure 3-3 shows a manual overlay of all gaze points onto a 2D screen view into the VR space. An automated Python script was developed to extract gaze events corresponding to Light ON and Light OFF states. Correlations were then established between the Response Time (time for user to notice and extinguish the light) and the baseline marker of visual attention (distance) as well as subsequent visual metrics such as Total Gaze Path travelled. Machine Learning models were then used to understand the contributions of distinct 2D visual markers to psychophysical behaviors like response time and attentional shift. The full method is detailed in Chapter 4: 2D Attention Geometry.

3.2.2 3D General Methodology

This thesis introduces a novel methodology for continuous visual attention mapping in motion. Attention orientation in three dimensions is extracted from original parameters and measures are derived which further represent gaze behaviors of interest.

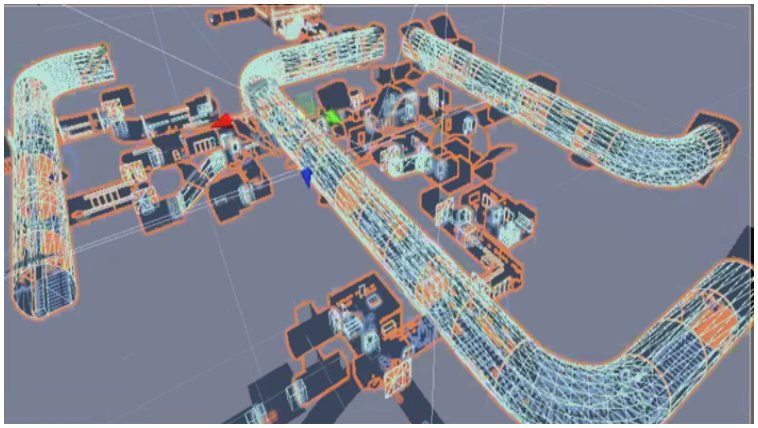


Figure 3-4. 3D Coordinates Simulation View (Zaman, 2020)

Three vectors were calculated from the position of the head as a local point of origin towards areas of interest (Figure 3-5). The Gaze Direction vector indicates the participant's line of sight, the Head Direction vector indicates the orientation of the head of the participant, and the Path Vector indicates the participant's direction of walking.

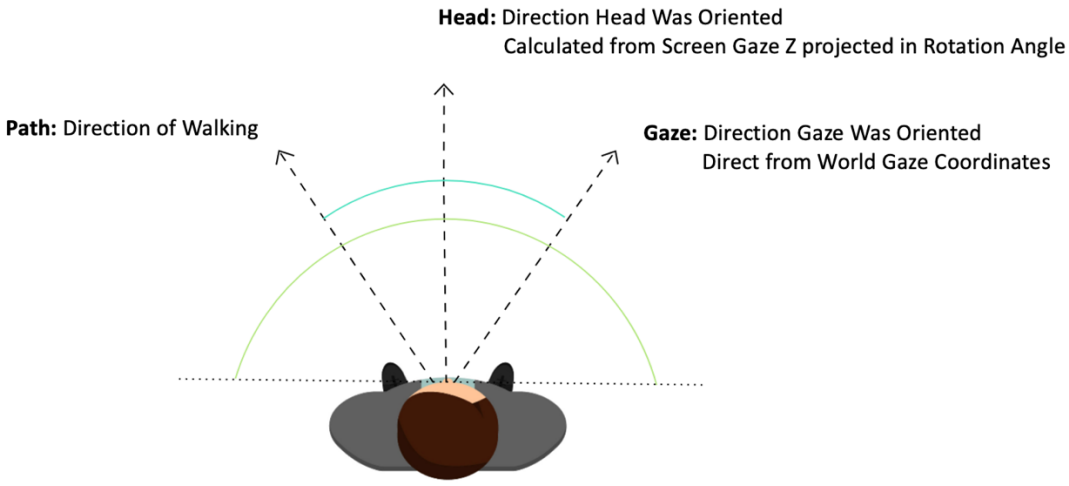


Figure 3-5. 3D Vectors: Gaze, Head, Path

Note: vectors angles adapted from Adobe Stock. WinWin (2024). People Top View. Male and female characters from above. [Vector illustration set]. https://stock.adobe.com/search?k=top+of+head+icon&search_type=usertyped&asset_id=413314791

The trigonometric functions used to derive the vectors and associated angles are defined in detail in Chapter 5: 3D Attention Geometry. In the absence of a static coordinate system in 3D space, dynamic points of reference were calculated relative to the user's position in space. A Python algorithm was developed to adapt gaze vectors continuously with the participant's movement. The participant's line of sight was mapped to the points in space where the gaze intersected a solid object. This method was developed as part of this thesis work and has not been documented in prior publications.

Figure 3-6 shows a high level summary of all derived features that are isolated as markers of 3D attention.

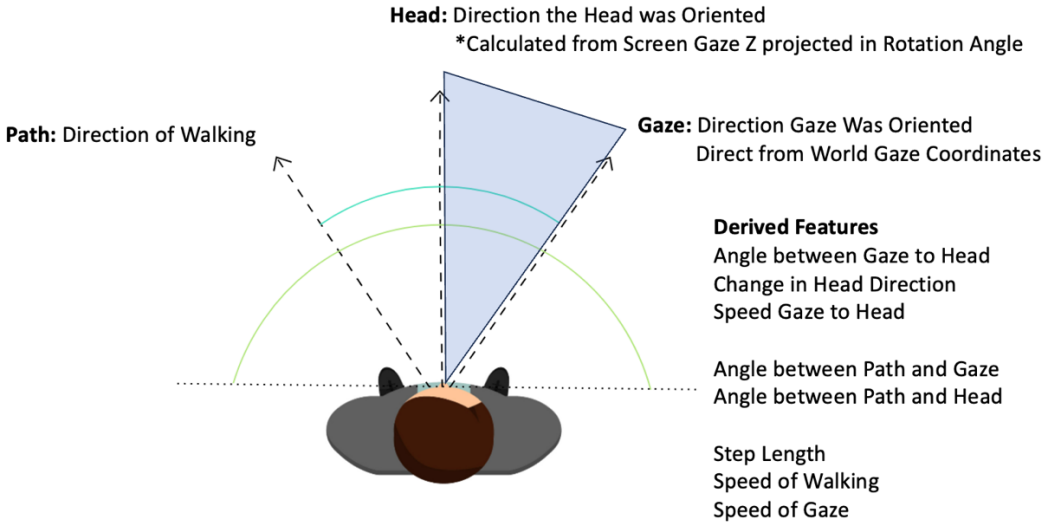


Figure 3-6. Summary of 3D Derived Features

Note: vectors angles adapted from Adobe Stock. WinWin (2024). People Top View. Male and female characters from above. [Vector illustration set]. https://stock.adobe.com/search?k=top+of+head+icon&search_type=usertyped&asset_id=413314791

This novel method and all derived features were used to answer Research Questions in the area of Decision-Making. Visual attention patterns were studied in several time-widows before and after users made a decision to ask for support. Machine learning models were then trained on the data from 48 participants to predict the points at which users feel lost and in need of navigational support.

3.3 Experimental Design Diagrams

3.3.1 Response Time

The first study in this thesis explores the relationships between several features of 2D visual attention and cognitive load, measured by the latency in response time to the light task.

Question 1-1: To what extent is there a correlation between gaze-to-stimulus distance and task response time latency?

The following experimental diagram details the null and alternative hypotheses which guide this study, as well as the two experimental studies performed: (1) The Preliminary One-Subject Study and (2) The 20-subject group study.

Table 2. Experimental Design 1: Distance and Response Time

RA 1-1: Gaze-stimulus distance and task response				
H_1(o): No correlation exists between gaze-to-stimulus 2D geometric distance and task response time. **H_1(a):** If the stimulus location lies further in the field of view from the visual center of the gaze, the time it takes to respond to the stimulus is longer.				
Independent Variable:	Distance from Gaze Center to Visual Stimulus on 2D Screen Plane (pixels)			
Dependent Variable:	Response Time (Time to Extinguish Light) (milliseconds)			
Study 1: (One Subject)	**Event 1**	**Event 2**	**. . .**	**Event N**
	nn Gaze Points	nn Gaze Points	**. . .**	nn Gaze Points
Study 2: **Group Study**	**Subject 1**	**Subject 2**	**. . .**	**Subject 20**
	52 Light Events	nn Light Events	**. . .**	nn Light Events

Question 1-2: To what extent is there a correlation between gaze travel and response time latency?

Table 3. Experimental Design 2: Gaze Travel

RA 1-2.: Gaze path length vs. task response time				
H_1(o): No correlation exists between gaze path length and task response time. **H_1(a):** As the response time to the stimulus increases, the gaze path travel length increases.				
Independent Variable:	Total gaze path movement during light task event (pixels)			
Dependent Variable:	Response Time (Time to Extinguish Light) (milliseconds)			
Preliminary Study (One Subject)	**Event 1**	**Event 2**	**. . .**	**Event N**
	nn Gaze Points	nn Gaze Points	**. . .**	nn Gaze Points
Group Study	**Subject 1**	**Subject 2**	**. . .**	**Subject 20**
	52 Light Events	nn Light Events	**. . .**	nn Light Events

Question 2: To what extent can a machine learning model be trained to accurately predict response time classes based on features of visual attention derived from eye movement data?

Table 4. Experimental Design 3: ML Predicting Response Time

RA 2: Machine Learning Accuracy, Response Time				
H_2(o): A machine learning model cannot accurately distinguish between fast and slow response. H_2(a): A machine learning model can be trained to accurately distinguish fast from slow responses.				
Independent Variable:	Total gaze path length during light task event (pixels) Distance, Gaze to Stimulus (pixels) Gaze Dispersion, X and Y (pixels)			
Dependent Variable:	Response Time (Time to Extinguish Light) (milliseconds)			
Target Variable Classes:	Class 1: Response Time **Below Threshold** Class 2: Response Time **Above Threshold**			
Target Class Thresholds:	Threshold 1: 1000ms Threshold 2: 2000ms			
Preliminary Study (One Subject)	**Event 1**	**Event 2**	**. . .**	**Event N**
	nn Gaze Points	nn Gaze Points	**. . .**	nn Gaze Points
Group Study 1	**Subject 1**	**Subject 2**	**. . .**	**Subject 20**
	52 Light Events	nn Light Events	**. . .**	nn Light Events

3.3.2 Attention Shift

Question 3: To what extent does attention shift towards the stimulus during a light event?

Table 5. Experimental Design 4: Attentional Shift

RA 3: Predicting attentional shift				
H_3(o): Attention does not shift relative to the stimulus over the course of the light event. H_3(a): Attention shifts towards or away from the stimulus over the course of the light event.				
Independent Variable:	Movement Direction (End Position - Start Position)			
Dependent Variable:	Attentional Shift			
Target Variable Classes:	Class 1: Gaze moved **towards** the stimulus Class 2: Gaze moved **away** from the stimulus			
Preliminary Study (One Subject)	**Event 1**	**Event 2**	**. . .**	**Event N**
	nn Gaze Points	nn Gaze Points	**. . .**	nn Gaze Points
Group Study	**Subject 1**	**Subject 2**	**. . .**	**Subject 20**
	52 Light Events	nn Light Events	**. . .**	nn Light Events

3.3.3 Light ON/ Light OFF

Question 4: To what extent can machine learning models distinguish the presence of the secondary task (LightON) from the LightOFF events based on features of visual attention derived from eye-tracking data?

Table 6. Experimental Design 5: ML Predicting Light ON / OFF

RA 4: Differentiating Light ON vs. Light OFF events				
$H_4(o)$: A machine learning model cannot accurately distinguish between LightON and LightOFF events. $H_4(a)$: A machine learning model can be trained to accurately distinguish between LightON and LightOFF events.				
Independent Variable:	All 2D Features			
Dependent Variable:	Attentional Shift			
Target Variable Classes:	Class 1: Light Stimulus is Present (Light ON Events) Class 2: Absence of Light Stimulus (Light OFF events)			
Preliminary Study (One Subject)	**Event 1**	**Event 2**	**. . .**	**Event N**
	nn Gaze Points	nn Gaze Points	**. . .**	nn Gaze Points
Group Study 1	**Subject 1**	**Subject 2**	**. . .**	**Subject 20**
	52 Light Events	nn Light Events	**. . .**	nn Light Events

3.3.4 Decision-Making

To understand visual markers of decision-making, visual attention patterns are analyzed over several time windows before and after the user calls for navigational support.

Question 5: To what extent can machine learning models predict the user need for navigation support from features of visual attention in 3D space derived from eye movement data?

On several occasions throughout the trial, the NavTools button was depressed through direct action by the user, displaying the NavTools map on the Heads-Up Display (Zaman et al., 2020). The NavTools-activation event is indicative of a user perception that he/she needs additional information to navigate further into the trial, followed by the user decision and action taken to display the information. This event provides a measurable signal. Four time-windows **before** and **after** the decision point are isolated and studied for attention markers of decision-making.

Table 7. Experimental Design 6: ML Predicting Decision Point

RQ 5: Predicting Decision Point				
H_5(o): A machine learning model cannot accurately predict when a user needs navigation support based on visual attention features. All models trained perform no better than random chance. **H_5(a):** A machine learning model can be constructed and trained on visual attention features to accurately predict when a user needs navigation support.				
Independent Variable:	All 3D Native Features, 3 Derived 3D Features: angle of rotation, world gaze			
Dependent Variable:	Decision Point Window			
Target Variable Classes:	Class 1: Before Decision Point Class 2: After Decision Point			
Target Class Boundary:	1: 30second window 2: 3 second window 3: 7 second window 4: 15 second window			
Group Study 1	**Subject 1**	**Subject 2**	**. . .**	**Subject 48**
	nn NavTools Events	nn NavTools Events	**. . .**	nn NavTools Events
Group Study 2	**Subject 1**	**Subject 2**	**. . .**	**Subject 48**
	nn NavTools Events	nn NavTools Events	**. . .**	nn NavTools Events

3.4 Data Engineering

3.4.1 Original Eye Tracking Feature Set

In the original Dynamic Information Needs Analysis (DINA) studies performed by Zaman et al (2020), 48 human subjects performed a virtual walk-and-search activity through a total of four separate VR layouts. As part of their original experimental data produced, eye tracking files were generated using the HTC Vive Eye Pro system for each run. Biometric data, location coordinates, head and eye tracking data, and object fixation durations were collected for 48 subjects and eight controls from multiple instruments and stored in separate files (Zaman, 2022).

One separate eye tracking file was generated by the HTC Vive Eye Pro system for each individual run in the DINA studies (Zaman, 2022). For this thesis, a total of three (3) relevant file types are selected from the original data and associated with each "run": a primary eye-tracking file, a light events file, and a Navigational Tool Events file (Zaman, 2022):

1. Primary Files containing Eye Tracking data were analyzed for each subject.
2. Secondary Files containing Light Task data generate Light Events
3. Secondary Files containing NavTools data are used to generate Decision Events

As part of this thesis work, the author conducts an initial data exploration activity to build a numerical and visual understanding of the source eye tracking files (Zaman et al, 2021). Features of interest are then extracted, aggregated, and analyzed to gain new insights into mechanisms of visual attention.

3.4.2 Primary File

The *Primary Eye Tracking File* contains gaze data which is distributed over 24 columns (features) and thousands of rows. All data in all columns is numerical (Zaman, 2022).

Table 8. Native Features in Primary Eye-Tracking Raw File (Zaman, 2022)

Column	Features	Units
1 - 2	Time Stamp, Interval Duration	milliseconds
3 - 5	Head position (x, y, z)	pixels
6 - 8	Head Rotation, Euler (x, y, z)	degrees
9 - 12	Head Rotation, Quaternions (x, y, z, w)	radians
13 - 15	Screen Gaze Position (x, y, z)	pixels
16 - 18	World Gaze Position (x, y, z)	pixels
19 - 21	Pupil Diameter (left, right, average)	millimeters
23 - 24	Eye Openness (left, right, average)	percent

3.4.3 Secondary Event Files

Two secondary files were used to isolate meaningful events (Zaman, 2020):

(1) the light task file, and

(2) the nav tools file.

Table 9. Native Features in Secondary Event Files (Zaman, 2020)

Light Task Event Files		
Column	**Features**	**Units**
1	Time Stamp at Onset	milliseconds
2	Duration of the secondary task	milliseconds
3	Task Correctness (True / False)	n/a

Navigational Tools Event Files		
Column	**Features**	**Units**
1	Time Stamp at Onset	milliseconds
2	Duration of the secondary task	milliseconds
3	Actuation (TurnOn / TurnOff/ Error)	n/a

3.4.4 General Data Process

The following steps were used to prepare the data for correlation and feature engineering:

1. Identify columns/ features of interest in independent files
2. Data cleaning on independent files
3. Structure one flat file data lake for analysis with all parameters
4. Annotate and clean aggregate data lake

3.4.5 Native and Derived Features

Figure 3-7 details the high-level data preparation process for the Machine Learning analyses. A further breakdown of features is captured in all subsequent chapters as it pertains to each respective analytical process.

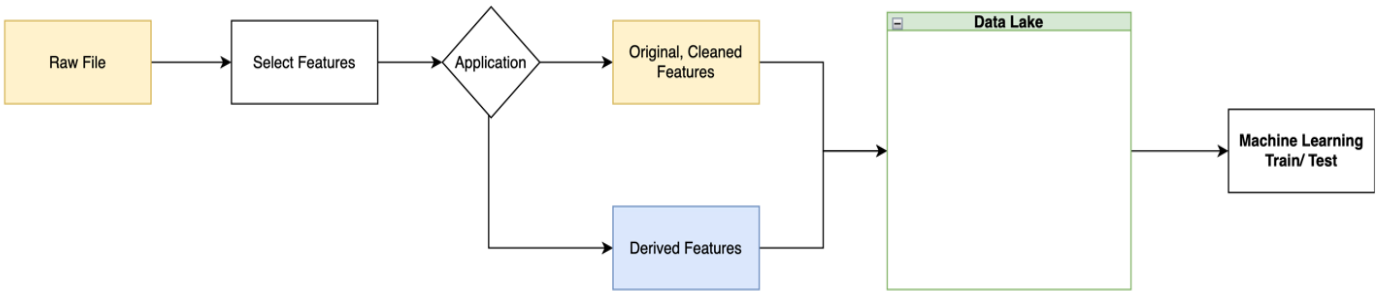


Figure 3-7. High Level Data Process

3.4.6 Feature Engineering over Multiple Datasets

The data process visualized in Figure 3-8 was used to extract and analyze features. Each subsequent chapter uses this data pipeline to structure data for statistical analysis and machine learning applications. The feature extraction process is depicted below. This method of data processing was consistent across the 2-dimensional and 3-dimensional analyses.

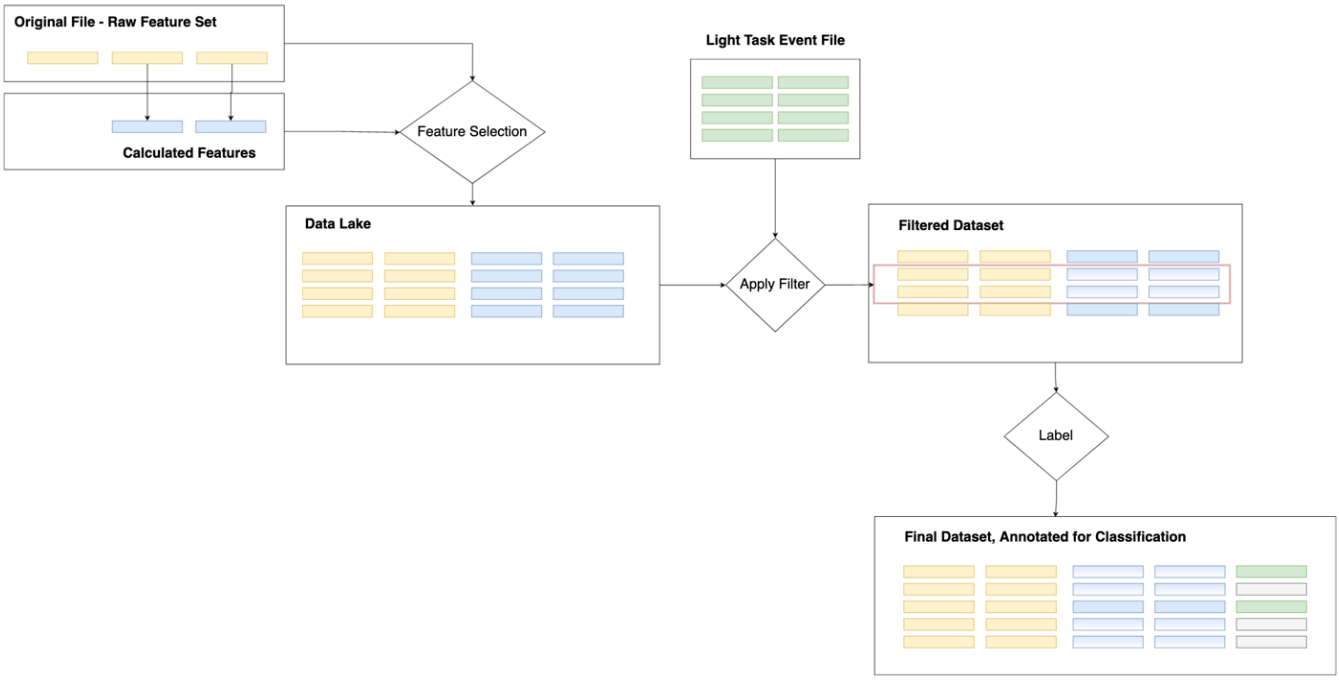


Figure 3-8. Feature Extraction Process

Features relevant to each analysis listed in the Experimental Design were selected from the original recorded eye tracking metrics (Zaman et al., 2020). For each analysis, new metrics were derived from the Primary File: an original eye gaze data file exported from the Tobii VR embedded eye-tracking SDK SRanipal. A data lake of original and derived measured was then compiled. Insights were extracted from the combined feature set. The initial analyses, which are aimed at characterizing visual attention allocation and gaze patterns in space and time, are all calculated on the curated data lake. A secondary set of analyses aim to identify cognitive and behavioral relevance from correlations to measured events. Temporal data from the Secondary Files were used to isolate meaningful groups of data corresponding to behaviors which were subsequently used to infer cognitive events. Features in the Primary File are represented in yellow, calculated features are represented in blue, and temporal behavioral events are depicted in green.

3.4.7 Data Cleaning

The following Protocol explains the Data Cleaning steps applied during data ingestion:

1. Imports two files:
    a. raw eye tracking data
    b. extracted event data (NavTools or Light Task)
2. Data Cleaning
    a. On Nav Tools File
        i. Removed entries where "extraInfo" column is set to TURNOFF
        ii. Removed entries where "extraInfo" column is set to USERERROR
        iii. Removed layouts where "Always On" condition was present
    b. On Light Task File
        i. Removed all rows with false light events from the light task file (button hit accidentally)
    c. Removed all gaze points outside of the bounds of the screen resolution.
        i. Removed all values where screenGazePosX < 0
        ii. Removed all values where screenGazePosX > 2880
        iii. Removed all values where screenGazePosY > 1600
        iv. Remove all values where screenGazePosY < 0
        v. Impute all values where screenGazePosZ < 0.02 with minimum
    d. Extra Columns:
        i. Dropped Quaternion Rotation Coordinates
    e. Remove Overlaps from Events
        i. If the difference between event timestamps is less than the machine-reported duration of an event, we remove the overlaps and re-index events

Null values were removed from Aggregate files of all 48 Participants for T-test and Machine Learning training.

3.4.8 Automated Data Processing

Python scripts were developed to automatically ingest the Eye Tracking Data and two Event Filters (Light Events for 2D and NavTools request Events for 3D). Automation scripts were developed in the Spyder environment, as this is a traditional Integrated Development Environment (IDE) with useful features like an integrated debugger and variable exploration. The general automation method is summarized in Figure 3-9.

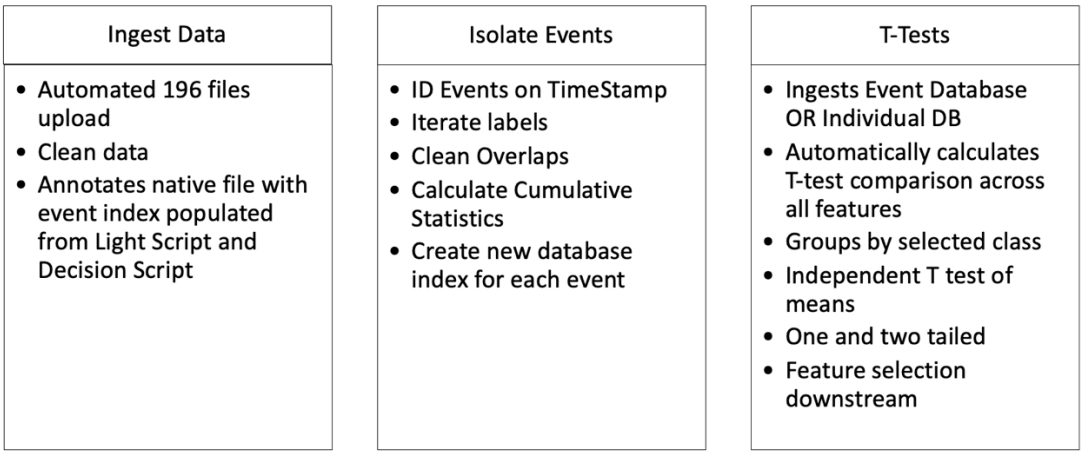


Figure 3-9. Summary of data automation scripts

3.4.9 Data Visualization

Figures in this thesis are generated using Python Seaborn, Matplotlib, and Plotly visualization packages. While most automation work was performed in the Spyder IDE, the majority of Python visualizations were generated using Jupyter notebooks. 2D and 3D visualizations of smaller datasets were generated in Python. These included examples of vector calculations, angles of rotation, and visual attention features reduced from one participant's data. For bigdata analysis, Python visualizations were used to show group-level statistics.

3D Visualizations of the 1.3-million data sets aggregated from all 48 participants were performed in VisIt, an open-source project. While VisIt was originally developed by the Department of Energy's Advanced Simulation and Computing Initiative (ASCI) to analyze tera-scale data, the tool is now used across industries for it's 2D and 3D visualizations features which extend to vector fields, adaptive meshes, and animations through time (Childs et al., 2012).

3.4.10 Machine Learning Training and Test Data

Machine learning algorithms are used to classify cognitive state against multiple biometric measures. Specific machine learning methods of inquiry were deployed using Scikit-learn libraries in Python. For all models below, the existing data are split into training and testing data. 70% of the data in the features of interest is randomly selected and allocated to the training set. A model validation phase focuses on hyperparameter turning and feature engineering for optimization of model accuracy, sensitivity, and specificity.

3.4.11 2D Datasets

Two types of datasets were aggregated from a combination of derived and native features. Extracted Events datasets contain events isolated for each analysis: "light events" isolated, indexed, and extracted for the 2D analyses, and "decision events" extracted for the 3D analyses. Table 10 shows the high-level summary of the 6 distinct data sets used for analyses in the chapters that follow.

Table 10. Aggregate Dataset Summary

	Datasets	Type	Size	# Subjects	# Layouts
1	**2D 1-Subject:** Light-Events Dataset " Biglights1"	Extracted Events	52 rows	1	1
2	**2D 1-Subject:** Indexed Raw Gaze Dataset " RawGaze1_Indexed"	Raw Gaze, Annotated	8087 rows	1	1
3	**2D 48-Subject:** All Light Events Dataset	Extracted Events	10,111 rows	48	4
4	**2D 48-Subject:** 2D Raw Gaze Points, Annotated Dataset	Raw Gaze, Annotated	1,336,996 rows	48	4
5	**3D 48-Subject:** Aggregate NavEvents Dataset	Extracted Events	2066 rows	48	4
6	**3D Angles, 1.3mil:** 3D Raw Gaze Points, Annotated Dataset	Raw Gaze, Annotated	1,336,996 rows	48	4

Figure 3-10 shows a small subset of the annotated raw gaze point dataset used in the 2-dimentional analysis. The features highlighted in Yellow below are Native Features (Zaman, 2022), extracted from the raw data files and cleaned as per the Cleaning Protocol (3.5.5). Each row depicted below indicates one gaze point measurement. In the 2D analysis, this includes the time stamp of each gaze motion, the screen gaze position in the X direction, and the screen gaze position in the Y direction (Zaman et al., 2021). The next two features, highlighted in green, are then calculated or "derived" features for the 2D individual raw gaze point dataset: the distance travelled from one gaze point to the next (*d_travel*), and the speed of each individual 2D gaze motion from one point on the screen to the next (*speed*). The final three features are annotated features used for classification. After the native features collected by Zaman et al. (2020) are processed through the Event Filter, each light event is assigned an index. The light index is then backpropagated onto the native features. All gaze points that occur in the absence of a secondary light task are annotated as -1, and the Light Status is labelled with the string "OFF". All remaining events are assigned a light index and the Light Status is labelled with the string "ON" (Zaman et al., 2020). The event filter iterates over light events and increases the index by 1 for each sequential event where a light-on is present. Finally, the Behavior feature classifies each gaze movement into Saccade or Fixation categories based on the speed of the movement.

	timeStamp	screenGazePosX	screenGazePosY	d_travel	speed	light	Behavior	LightStatus
4248	173220	1268.217	1074.763	33.09477161	0.7521539	-1	Saccade	OFF
4249	173269	1271.842	1079.537	5.994305715	0.12233277	28	Fixation	ON
4250	173322	1272.241	1082.73	3.217833122	0.060713832	28	Fixation	ON
4251	173378	1269.987	1095.402	12.87090129	0.229837523	28	Saccade	ON
4252	173424	1278.488	1094.888	8.516524937	0.185141846	28	Fixation	ON
4253	173467	1273.044	1094.817	5.444462967	0.126615418	28	Fixation	ON
4254	173519	1185.06	1109.513	89.2028961	1.71544031	28	Saccade	ON
4255	173569	1186.512	1107.198	2.732677991	0.05465356	28	Fixation	ON
4256	173614	1172.935	1105.44	13.69034306	0.304229846	28	Saccade	ON
4257	173665	1175.84	1124.75	19.5272918	0.382888074	28	Saccade	ON
4258	173722	1159.154	1091.642	37.07508948	0.650440166	28	Saccade	ON
4259	173782	1263.406	1076.277	105.3781891	1.756303151	28	Saccade	ON
4260	173829	1253.082	1082.996	12.31787064	0.262082354	28	Saccade	ON
4261	173890	1257.625	1087.99	6.751213595	0.110675633	28	Fixation	ON
4262	173934	1228.274	1005.176	87.86148074	1.996851835	28	Saccade	ON

Figure 3-10. Snapshot of Features in 2D Annotated Dataset

The specific breakdown of numerical features and categorical features (encoded for classification) for the 2D 1-subject datasets used in preliminary study are listed below.

Table 11. One-Subject, 2D Datasets used in Preliminary Study

	Dataset	Data	# Subjects - # Layouts	Data Count	Features (Numerical)	Classifier Features (Categorical)
1	**Biglights1:** G_Features_MacroLight_S01_ff	2D Light Events	1 - 1	52	1. timeStamp 2. timeDuration 3. distance 4. distance_start 5. distance_end 6. dst_delta 7. length 8. length_rate 10. GazeDispersionX 11. GazeDispersionY	9. position
2	**RawGaze1_Indexed:** G_Gaze_LightTaskIndex_S01_ff	Annotated Gaze Points	1 - 1	8087	1. timeStamp 2. screenGazePosX 3. screenGazePosY 4. distance_travel 5. speed	6. Light 7. Behavior 8. LightStatus

Note: Features highlighted in blue represent native values (Zaman, 2022). Black features represent new values derived via the data handling methods here. All data are processed through the cleaning filter.

The second data set includes parameters which describe cumulative event-level behavior calculated across one entire light task. To generate this data set for Subject 1, a Python script (The Event Filter) was written to extract the gaze behaviors between the start and end time stamp of each sequence where the light was ON. For this preliminary composite dataset, 52 continuous sequences were identified. Table 12 details the full set descriptive statistics.

Table 12. One-Subject, 2D Light Events

	timeStamp	timeDuration	dst	dst_s	dst_e	dst_d	lng	lng_rate	pos	GazeDispersionX	GazeDispersionY
count	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00
mean	175593.12	2033.12	1117.34	1128.64	1161.58	32.94	2565.81	0.03	0.42	107.89	66.28
std	108068.15	1943.56	103.64	143.89	172.51	171.31	2953.82	0.06	0.50	69.85	43.39
min	8184.00	458.00	904.12	881.68	709.09	-227.29	144.25	0.00	0.00	5.23	10.07
25%	79847.75	774.25	1040.00	1019.29	1050.74	-59.18	606.36	0.00	0.00	52.55	36.48
50%	168563.00	1014.00	1119.39	1129.38	1132.22	6.99	1264.98	0.01	0.00	93.58	63.54
75%	273912.50	2436.00	1187.27	1222.37	1255.29	82.62	3101.00	0.03	1.00	157.87	83.70
max	344424.00	8263.00	1412.71	1440.04	1777.31	617.19	10602.12	0.45	1.00	356.71	291.19

A second python script was written to back-propagate the light task indexes isolated for each event onto the raw gaze data. Light-OFF events were annotated with the number -1, and all subsequent continuous Light-ON events are iterated upwards throughout a full trial. Table 13 highlights the descriptive statistics of the numerical fields in the final annotated 2D 1-subject gaze point dataset used in preliminary studies.

Table 13. One-subject 2D Native Gaze Features, Light Index Annotated

	timeStamp	screenGazePosX	screenGazePosY	d_travel	speed	light
count	8087.000000	8087.000000	8087.000000	8086.000000	8086.000000	8087.000000
mean	171483.516384	1108.818069	1080.213795	57.591480	1.272780	6.282923
std	103822.338882	171.054109	121.605974	110.824624	2.333988	13.417637
min	22.000000	24.707130	91.715810	0.082098	0.002052	-1.000000
25%	79446.500000	1016.107000	1024.675000	7.750135	0.184421	-1.000000
50%	165045.000000	1113.293000	1081.542000	22.121860	0.520824	-1.000000
75%	260608.500000	1200.205000	1147.717000	59.905159	1.364134	8.000000
max	362891.000000	2639.059000	1547.466000	2025.707340	59.579628	51.000000

***RawGaze1_Indexed**

3.4.12 3D Datasets

The datasets in this section are aggregated from the original raw data for all 48 subjects (Zaman et al., 2021), cleaned and annotated by python scripts presented here. New visual attention markers are derived from the original dataset collected by Zaman et al. and aggregated, and the composites of the original data and processed features used to train Supervised and Unsupervised Machine Learning models in the analysis of 3D Gaze patterns.

Visual attention features in 3D are extracted for multiple timeframes before and after the user requests the NavTools information on the HUD. This NavTools request action is of interest as it is a marker of the user's decision that he/she needs further information to navigate further into the trial (Zaman et al., 2020).

Figure 3-11 presents the result of the 3D decision event feature annotation process. The time stamps of each decision event are isolated from one participant. 3D gaze path length information is calculated **before** the decision to request navigational support and **after** the information is presented. "Gaze Path Length Before" and "Gaze Path Length After" features are populated into two adjacent columns by a Python script. For the first analysis, Gaze Path Lengths are calculated within 30-second timeframes Before/After the NavTools request. For this participant, 10 events were identified and annotated with derived class features.

	timeStamp	TotPathLengthBefore	TotPathLengthAfter
0	49453	3392.732374	2189.407227
1	50467	3562.817195	2009.978875
2	261391	799.357963	443.005200
3	436895	826.244826	514.866227
4	437611	785.527442	507.179516
5	443060	662.013440	484.771612
6	487419	504.099272	869.323832
7	533569	624.206901	271.207066
8	597011	555.678005	689.209094
9	600830	517.296225	726.706009

Figure 3-11. Decision events annotated with Before/ After-Decision Gaze Path Length
Note: timeStamps extracted from Zaman (2022)

This method of data selection is used for all T-Test comparisons of means for each feature in the 'before' and 'after' condition. Two algorithmic Python methods were developed to process the T-test comparisons. The first method was more descriptive at the individual level, while the second method was significantly faster to execute across all features but lacked the nuance of the individual participants.

This data layout is useful for T-test analysis of means. Additionally, a separate script is written and executed to transcribe the Before and After conditions into one column for every feature of interest. This latter method allows for all "classes" to be encoded in the same column for machine-learning model training.

Table 14 shows descriptive statistics for all 3D World Gaze Features derived according to the method above and compiled for all 48 subjects across 4 layouts.

Table 14. Full 3D Feature Set, Descriptive Statistics

	Native Features (Zaman, 2021)				Derived Features			
	timeStamp	headRotEulerY	avgPupilDiameter	avgOpenness	ChangeHeadRotation	Speed_HeadRotEulerY	AngleGazetoHead	AngleSpeed_GzHd
count	1.336996e+06	1.336996e+06	1.336996e+06	1.336996e+06	1.336996e+06	1.336996e+06	1.336996e+06	1.336996e+06
mean	2.334207e+05	1.819908e+02	5.281737e+00	9.134652e-01	4.475715e-03	3.261465e-02	8.338364e+01	-1.183104e-02
std	1.685542e+05	1.065807e+02	1.596516e+00	1.738528e-01	3.126424e+00	4.383298e+01	6.557828e+01	1.339710e+02
min	9.200000e+01	-5.540458e-03	-1.000000e+00	0.000000e+00	-1.687300e+02	-8.366242e+02	2.831647e-06	-3.625703e+03
25%	9.960600e+04	8.825498e+01	4.893072e+00	9.057874e-01	-3.232215e-01	-6.138011e+00	1.663616e+01	-1.992986e+01
50%	2.030245e+05	1.781282e+02	5.528511e+00	1.000000e+00	4.849500e-03	9.465153e-02	7.403910e+01	1.696744e-01
75%	3.298042e+05	2.697990e+02	6.154160e+00	1.000000e+00	3.586000e-01	6.750956e+00	1.533571e+02	2.038735e+01
max	9.314120e+05	3.599940e+02	9.688118e+00	1.000000e+00	1.443771e+02	8.126780e+02	1.800000e+02	3.045877e+03

All data which included negative values in the Z direction (into the unity world) were replaced with the minimum value of the dataset. Since it is not possible to “look” behind your head, it is assumed that negative values were incorrect recordings. The lowest value greater than 0 present in the dataset (0.02) was imputed to replace negative Z values.

Events were isolated in the time window of 30 seconds before and 30 seconds after the timestamp at which the user decided to request the Navigational support information. The features above were extracted, annotated into two classes (Before and After the decision point), and compiled for class-based supervised machine learning analysis.

3.5 Algorithms

3.5.1 Automated Data Ingestion Script

The pseudocode steps defined in Table 15 present the high-level logic of the automation script which is used to clean, aggregate, and isolate data slices from all participants in the original study into relevant events for classification (Zaman et al., 2021). The script further annotates meaningful classifications for each isolated event.

Table 15. Data Pre-Processing Pseudocode

Ingesting Data and Data Cleaning Script	
0:	Import Libraries
1:	Loop through all subjects
2:	Loop through all layouts
3:	Data Cleaning protocol
4:	Select event data slice
5:	Create empty fields for derived features
6:	Create temporary data frame to store filter
7:	Extract event-level statistics
8:	Assign class at event-level and backpropagate to original file
9:	Output annotated gaze features
10:	Output annotated events

3.5.2 Logistic Regression Script

The pseudocode in Table 16 outlines the algorithmic steps used to run Logistic Regression Machine learning models in Python. Similar steps are taken to train and test the remaining supervised models described in this thesis.

Table 16. Machine Learning Script Pseudocode

	Logistic Regression Script	
1:	**Input 1:** 1 Subject - Gaze Dataset **Input 2:** 1 Subject - Light Events 2D Dataset	
	Data Cleaning and Processing	
2:	Remove rows with null values	*gaze.dropna()*
3:	Split response variable into bins based on manually defined threshold	*np.digitize()*
4:	Find categorical data in gaze	*if gaze[column].dtype == object and len(gaze[column].unique()) <= 50:*
5:	Encode categorical variables	*label = LabelEncoder(), label.fit_transform(gaze[column])*
	Train Logistic Regression Model	
6:	Set data: 70% to train, 30% to test	*X_train, X_test, y_train, y_test =* ***train_test_split*** *(X1, Y1, test_size=0.3, random_state=0)*
7:	Instantiate Logistic Regression Object	*logreg =* ***LogisticRegression()***
8:	Fit the model on our training data	*logreg.**fit**(X_train, y_train)*
9:	Make predictions on Testing Model	*y_pred = logreg.**predict**(X_test)*
	Model Evaluation	
10:	*Create Confusion Matrix*	cm = confusion_matrix(y_test,y_pred)
11:	Calculate logistic model intercept	model.intercept_
12:	Calculate logistic model coefficient	model.coef_
13:	Calculate model accuracy	Accuracy = TP+TN/(TP+TN+FP+FN)
	Visualize Logistic Regression Model:	
14:	Set sigmoid function using expit()	*sigmoid_function =* ***expit****(X1 * model.coef_ + model.intercept_)*
15:	Plot sigmoid function	*plt.**scatter**(X2, sigmoid_function)*

CHAPTER 04:

2D ATTENTION GEOMETRY

4 2D ATTENTION GEOMETRY

The study of 2D attention uses projections of eye gaze fixations onto the 2D screen. The point where the eye gaze vector intersects the Display Area (z coordinate = 0) is calculated by the Tobii software. Figure 4-1 shows the developer coordinate system that orients the screen gaze positions used in this thesis work. Of note, the origin of the system lies in the top right corner of the screen and is considered as such when the coordinates of the light stimulus are determined.

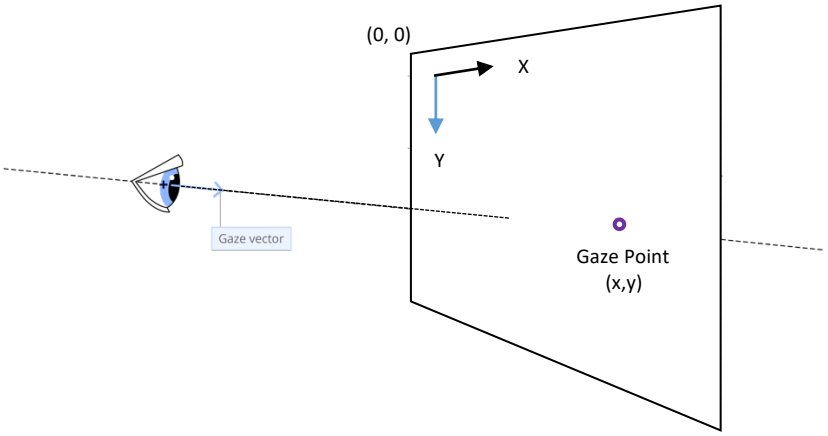
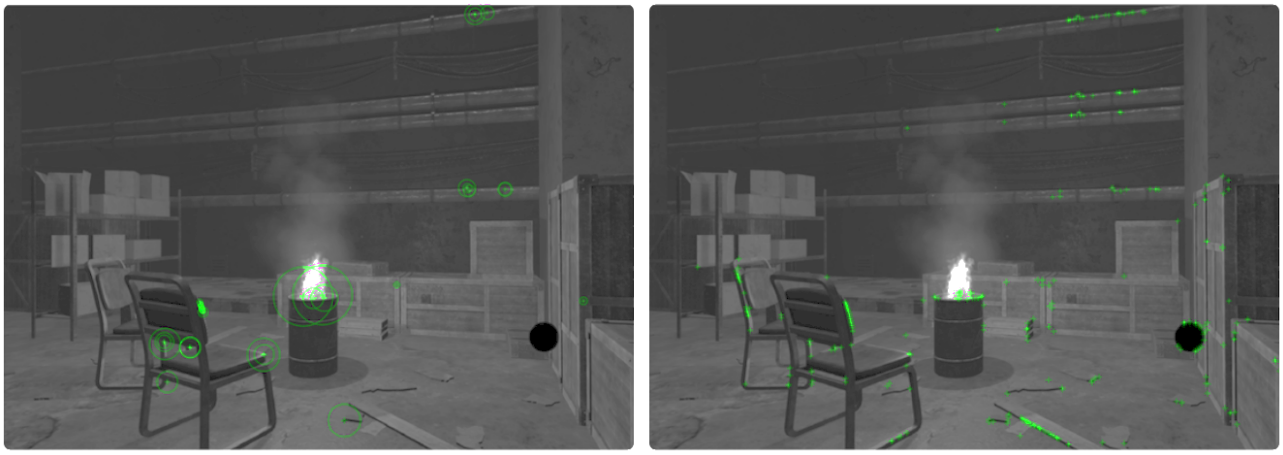


Figure 4-1. Gaze Vectors Onto 2D Screen Coordinates
Note: Adapted from Tobii AB, 2024. https://developer.tobiipro.com/commonconcepts/coordinatesystems.html

MATLAB edge detection algorithms were used to locate and extract the coordinates of the cognitive task light. Images were extracted from original video of the VR Unity simulator (Zaman et al., 2020). The images were then processed in MATLAB, and two edge detection models were executed to identify bright areas and corners in low luminescence scenarios. The results are presented in Figure 4-2. The Brisk Feature Extraction algorithm was not successful at identifying the light location as it focused on locating objects based on brightness (Figure 4a). The Harris Point Corner Extraction feature successfully detected the edges of the light task. Once extracted, Harris features can be individually selected, and light coordinates identified.



(a) Brisk Feature Extraction (b) Harris Point Corner Extraction

Figure 4-2. MATLAB edge detection models (adapted from Zaman, 2022)

4.1 2D Geometric Features

4.1.1 Fixation Eccentricity

Figure 4-3 depicts a composite of gaze fixation coordinates and field of view inside the subterranean simulator. The geometric distance between the centers of the gaze cluster and the cognitive load light task are visualized onto the first-person 2D Screen Plane view.

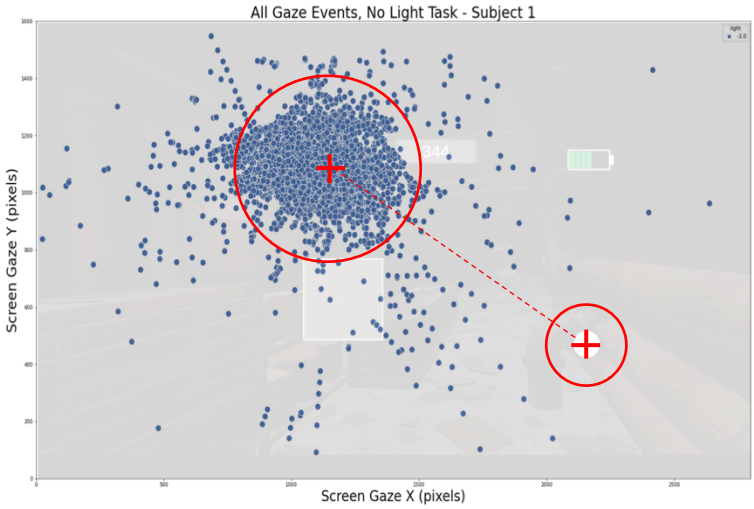


Figure 4-3. Geometric distance between fixation centroid and HUD cognitive load light

Fixation x- and y- coordinates are recorded in pixels. From the HTC Vive Pro manufacturer's specs, the combined resolution of the field of view for this setup is 2880 x 1600 pixels. The center of each fixation centroid is recorded in pixels and compared to the center of the light task coordinates. This schematic is an example of all gaze location recorded for one subject across one layout, manually overlayed onto a static image of the VR sub-T space. This serves as a conceptual demonstration of the relationship between gaze fixation and the light stimulus. Fixation locations are extracted for all subjects across all light events. A study is conducted to identify correlations between the patterns of visual attention including this gaze-to-stimulus distance, gaze length travel over time, gaze dispersion, and response time latency.

4.1.2 Distance

The distance formula in two-dimensional space is calculated:

$$d_{g-s} = \sqrt{(d_x)^2 + (d_y)^2}$$

where the x- and y- distances are derived for each iteration in gaze location:

$$d_x = ((x_{ts_{screen}} - x_{(ts-1)_{screen}})^2$$

$$d_y = ((y_{ts_{screen}} - y_{(ts-1)_{screen}})^2$$

$$d_{g-s} : Distance\ Between\ Gaze\ and\ Stimulus$$

and individual coordinates in the horizontal and vertical direction are extracted from raw data :

$x_{n_{screen}}$: $Horizontal\ Screen\ Gaze\ Position\ at\ Gaze\ time, ts$

$y_{n_{screen}}$: $Vertical\ Screen\ Gaze\ Position\ at\ Gaze\ time, ts$

ts_{gaze} : $Timestamp\ of\ Gaze\ Event$

4.1.3 Gaze Dispersion

Gaze dispersion for Subject 1 is visualized before and after Cleaning using x- and y- distribution based on a technique described by Louw et al (2018). Data points beyond 2 standard deviations from the mean are removed to obtain a fixation cluster. Plotly libraries in Python are used to generate the gaze dispersion plots in Figure 4-4. This method is used to understand the gaze dispersion in vertical and horizontal directions (Louw, 2018).

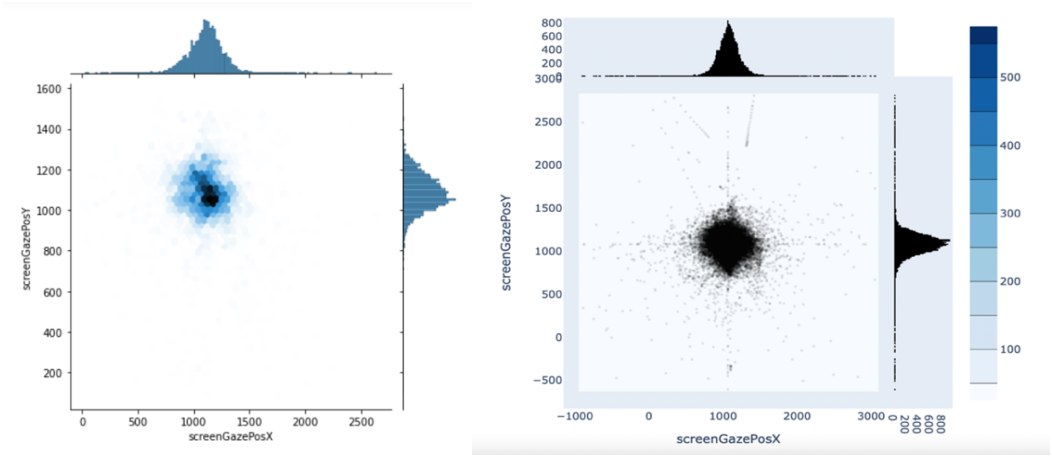


Figure 4-4. Visualizing horizontal and vertical gaze dispersion for one participant

4.1.4 Saccade and Fixation

Two visualizations depict the screen gaze position in the horizontal direction X direction (Figure 4-5) and vertical direction Y (Figure 4-6) across all time stamps for Subject 2 across one run in the original study (Zaman et al., 2021).

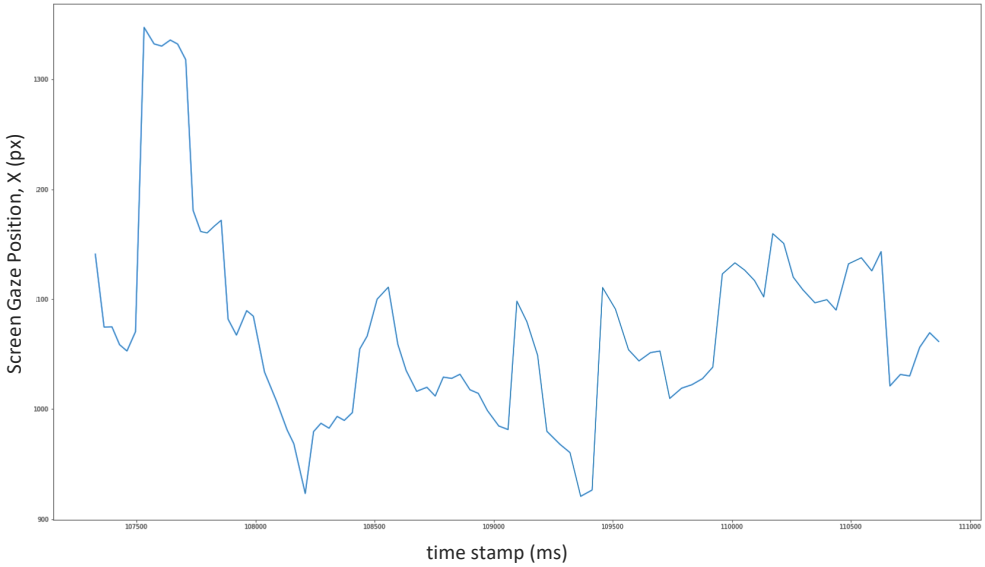


Figure 4-5. Saccades and Fixations in X- direction independently across one light event

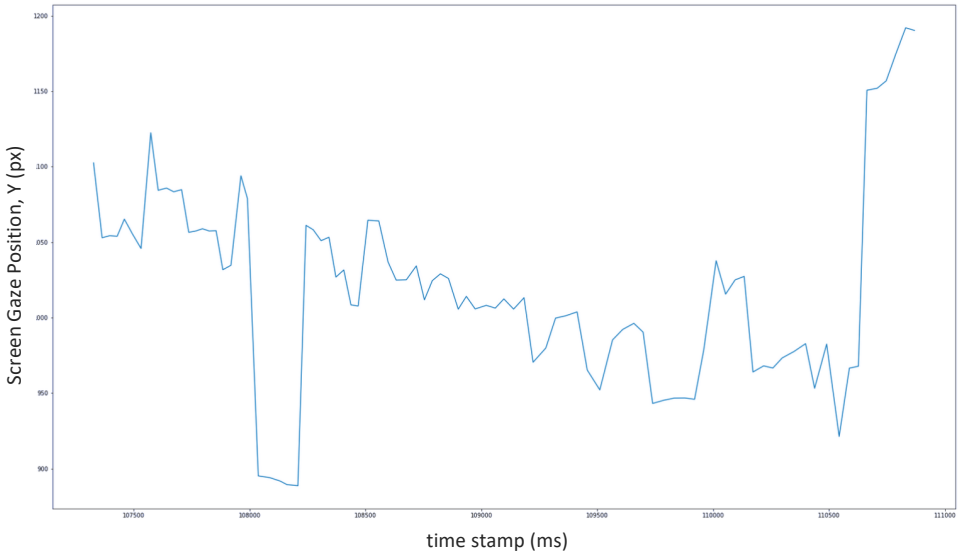


Figure 4-6. Saccades and Fixations Y- direction independently across one light event

4.2 Isolating Events

4.2.1 Light-On vs. Light-Off Events

Figure 4-7 plots the screen gaze coordinates in the X and Y directions for one subject at all time points when the light was off.

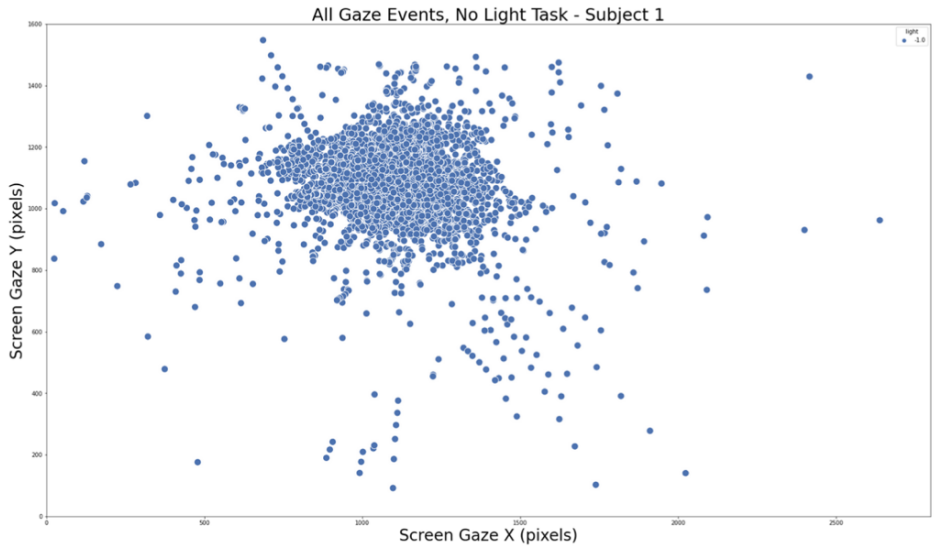


Figure 4-7. Gaze distributions for light-off events isolated for one subject.

Gaze events pertaining to each indexed Light-ON time window were extracted. Gaze points corresponding to each independent light event are separately color-coded in Figure 4-8. The legend shows all light event numbers and associated colors isolated for one subject run.

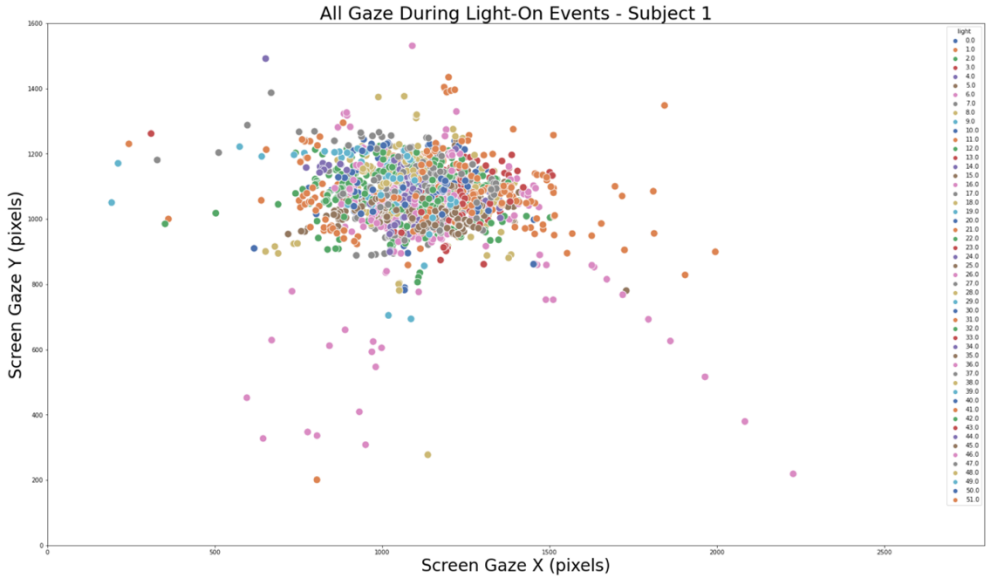


Figure 4-8. Gaze distributions for light-off (a) and light-on (b) events for one subject

An interesting observation is that gaze dispersion is comparable for the two groups of events in the x-direction, but the gaze dispersion in the y-direction is smaller during the LIGHT-OFF periods as compared to the periods of search in the absence of the secondary light task. In other words, subjects continued looking in the same horizontal pattern when the Light Distraction task was ON as when it was OFF. However, in the vertical, or up-and-down direction, there was a noticeable difference in the pattern of eye movements, as seen in the difference between Figures 4-7 and 4-8.

4.2.2 Isolating Light Task Events

Next, once light events are isolated from the data, timestamps of all gaze coordinates for the duration of one event are depicted onto the x-y coordinates using a color gradient. As seen in Figure 4-9, increasing time corresponds with darker coloration. Gaze points at the beginning of the light task are lighter in shade, while gaze points towards the end of the gaze path are darker.

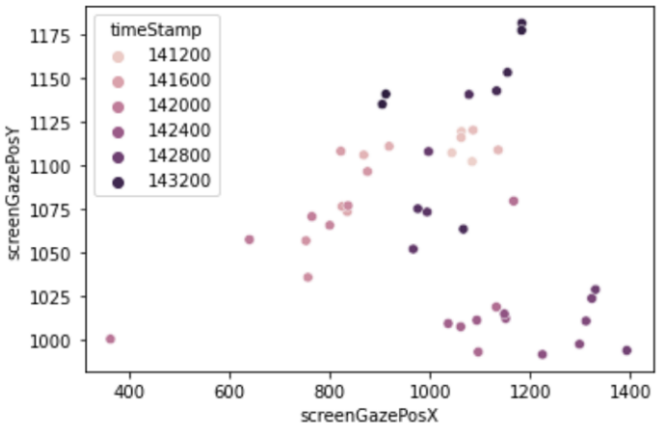


Figure 4-9. Eye movements isolated light event tracked by time gradient.

4.3 Extracting Light Sequences

The gradient visualization method facilitates the extraction of sequences of events, as depicted in Figure 4-10. Here, the participant gaze during a sequence of four light-task events are displayed across time. For each event, the gaze path can be followed from the time the light turns ON (light, pink) until the time when the light is extinguished (dark, purple).

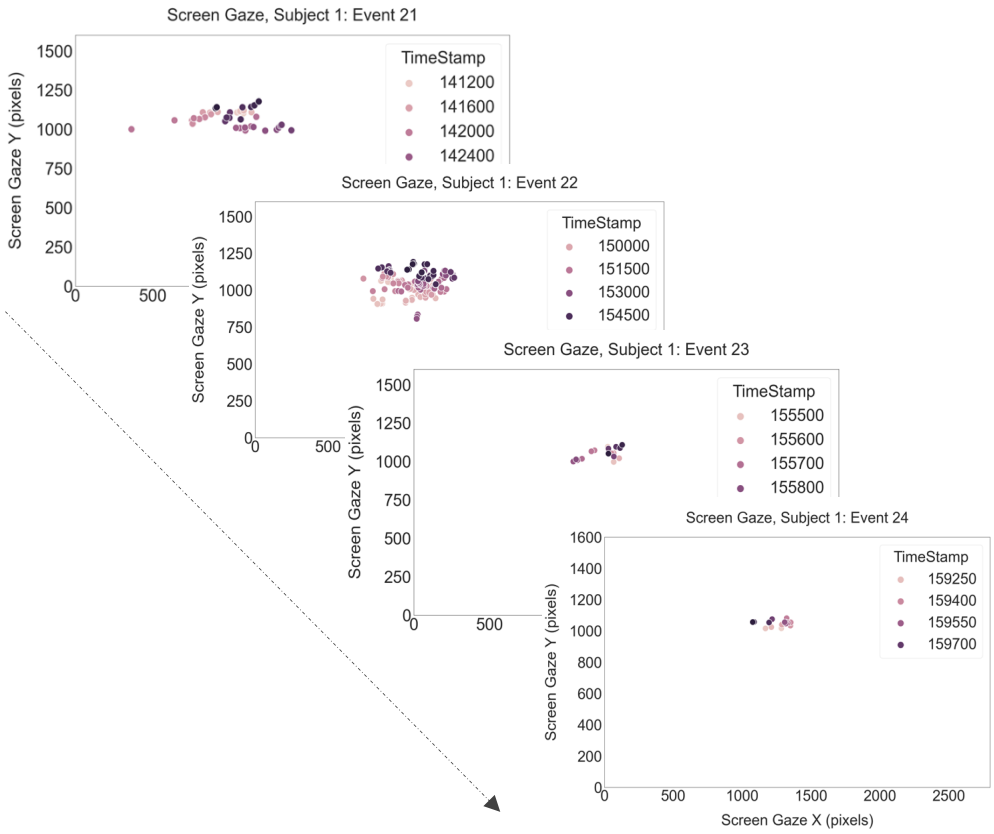
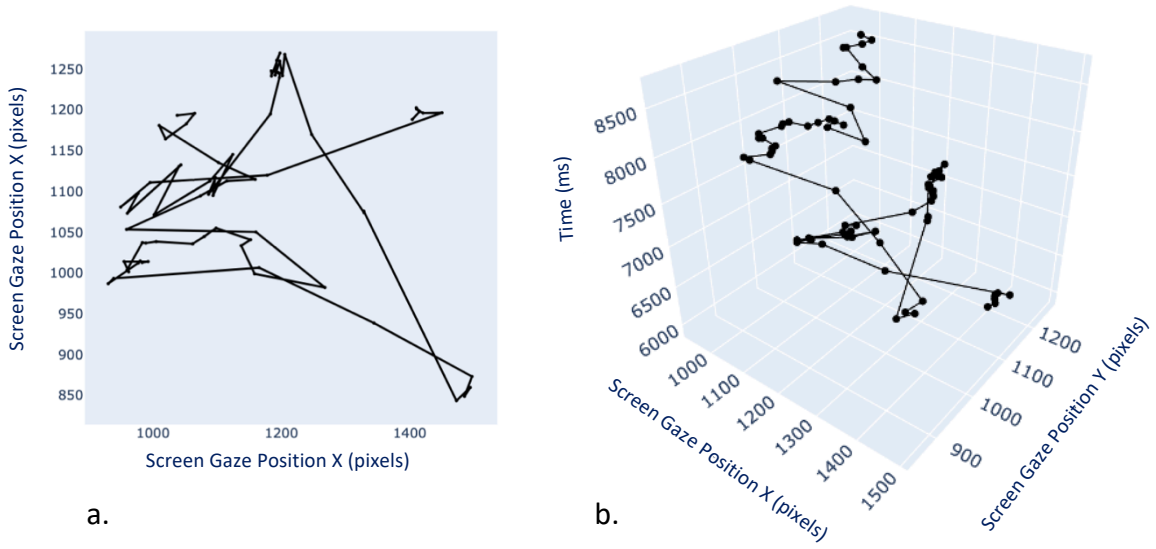


Figure 4-10. Light Task Event Sequence over Time

4.4 Space-Time Domain

Figure 4-11 shows the screen gaze positions for all gaze points for one light task event for one subject during one light task event (a). The gaze path for one event is shown as the X-Y coordinates on screen, where position coordinates are defined using X and Y pixels. The gaze trajectory is visualized in the space-time domain for the duration of the light event, where time is represented along the Z axis (b).



a. b.

Figure 4-11. The gaze path for one event in XY plane (a) extended into the time domain (b)

4.5 Correlations

Correlation analyses were performed on the two preliminary datasets generated (1) the Annotated Raw Gaze (Figure 4-12) and Light Events 2D Features (Figure 4-13).

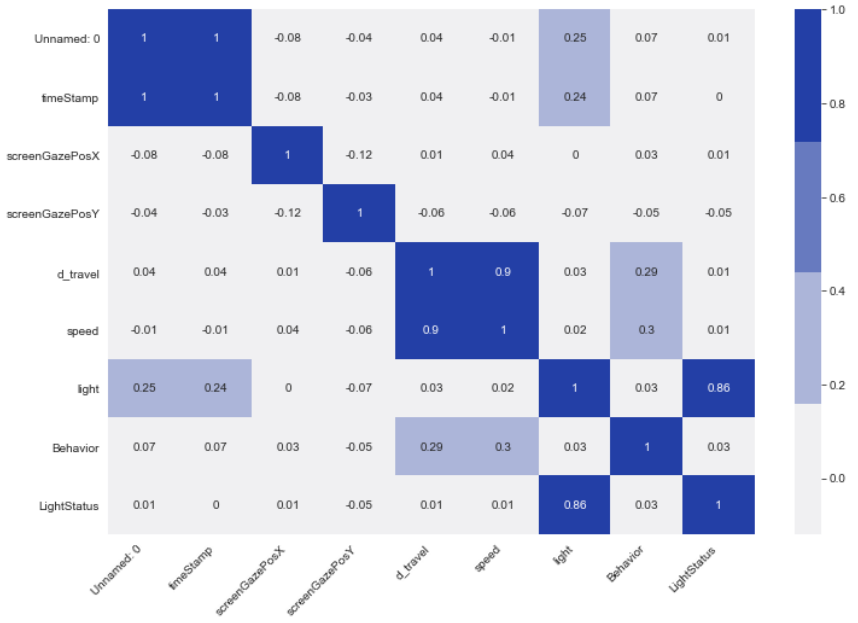


Figure 4-12. Correlation Plot of Annotated Raw Gaze and Features

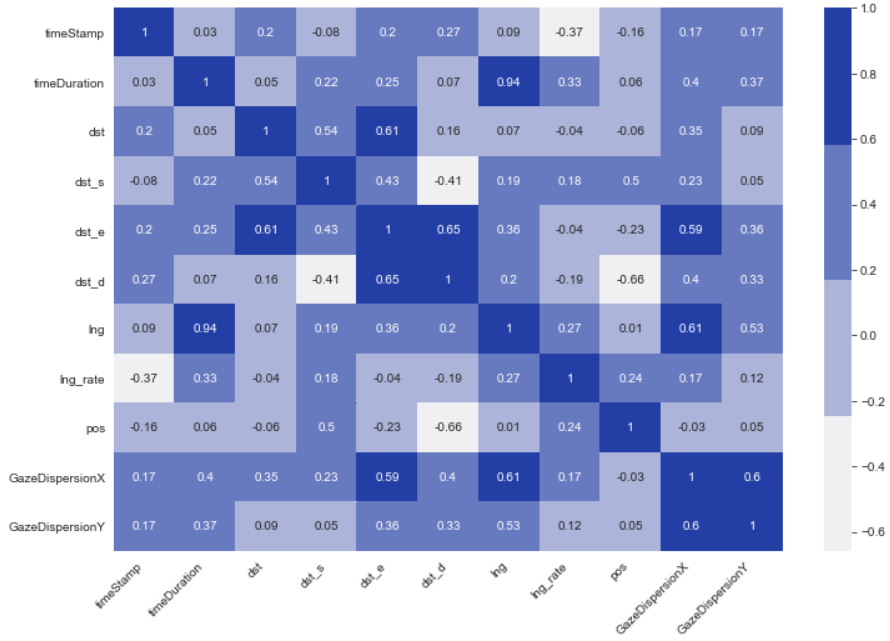


Figure 4-13. Correlation Plot of Light Event Features

CHAPTER 05:
3D ATTENTION GEOMETRY

5 3D ATTENTION GEOMETRY

This chapter introduces 3-Dimentional Features and explains the geometric and trigonometric transformations

- Can we predict the moment when participants were lost?
- We want to understand attention patterns while in motion from eye movement and head movement behaviors
- Is there a correlation between visual attention parameters derived from eye tracking readings in three dimensions and the moment participants request map data?
- What patterns lead up to the moment when the user requests navigation data?

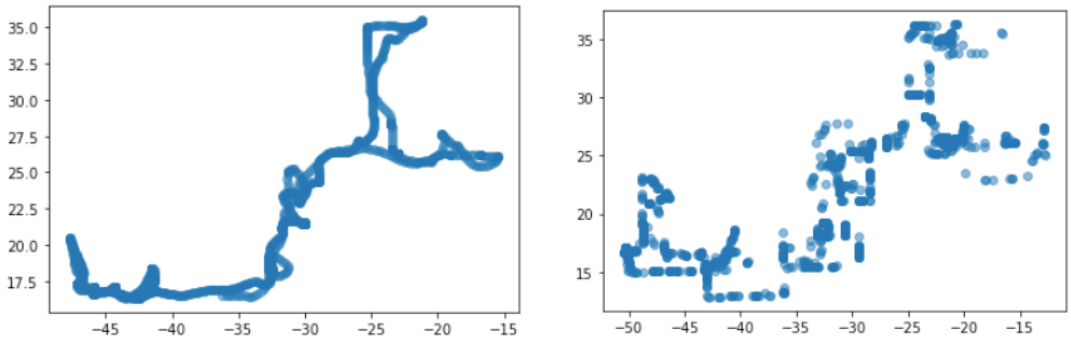
- Understanding the Features
    - Working with Euler Angles
    - World Gaze Coordinates
    - Visualizing Gaze Vectors in World View
    - Data Cleaning across Euler Angles
- Calculated Features
    - Calculating Visual Angle
    - Calculating Angular Velocities
    - Extracting NavTools Decision Events
- Results
    - Qualitative Observations
    - Quantitative Results
    - Predicting Decision Points

5.1 3D Features in AR/VR Space

The previous chapters rely on the 2-dimensional **Screen Gaze Coordinates** (x, y), where the gaze intersected the Screen Plane in AR/VR space. In this chapter, screen coordinates in the Z direction are utilized to understand gaze patterns beyond the screen and "into" the Unity VR world. World Gaze Position (x, y, z) is the location in the Unity world (in meters) where the gaze landed at all points in the trial (Zaman et al., 2021). The **Head Rotation, Euler** (degrees) and the **Head Position (x,y,z)** are also mapped for 3D attention representation. The "z" direction stretches forward into the unity tunnel world. When viewed from "above", the **Head Position Z** outlines the tunnel map. These metrics are used in this chapter to determine the "paths" of visual attention and walking.

5.2 Head Position and Gaze

Head position in X and Z dimensions is visualized (Figure 5-1a) alongside the World Gaze Position coordinates in the 3D space (Figure 5-1a). The head position indicates where subjects were standing, while the world gaze position indicates where the gaze landed. In later analyses, these features are overlayed on the same trace to visualize gaze vectors. Both images align with the map of the subterranean environment which the participant traversed during one trial.

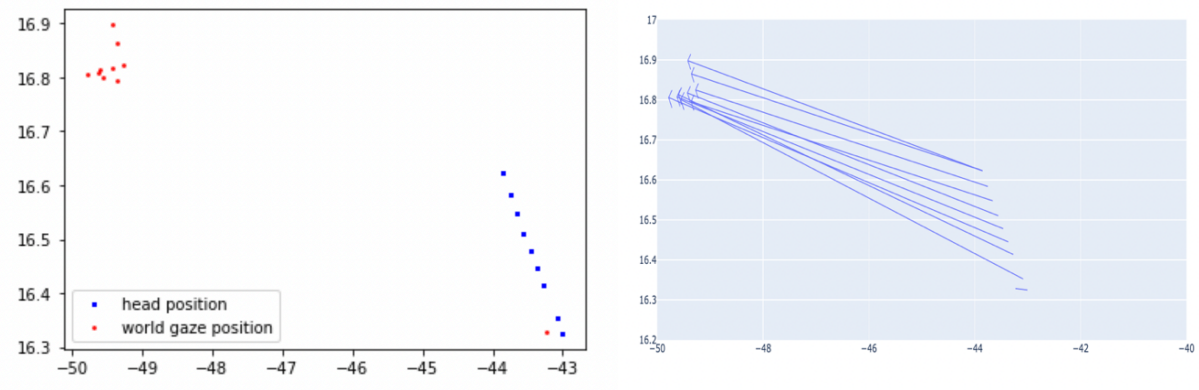


(a) (b)

Figure 5-1. The Head Position X vs. Z in meters (a) and world gaze position (b)

5.3 Gaze Vectors

Figure 5-2(a) shows a closer look at 10 gaze readings. The participant is walking in a straight line, indicated by the head position coordinates in Blue. The gaze is fixated on a cluster of points indicated by the red world gaze position coordinates. Figure 5-2(b) shows the resulting vectors extending from the head position to the world gaze position. This method is used to visualize gaze coordinates in subsequent analyses in this chapter.



(a) (b)

Figure 5-2. World Gaze and Head Position - individual points (a) and vectors (b).

Five raw variables (features) are extracted for the 3D gaze analysis. These features are original readings, observed on the day of the experiment and captured by the Tobii eye tracker in the original data (Zaman, 2020).

1. x_{head}: $Head\ Position\ X$
2. z_{head}: $Head\ Position\ Z$
3. x_{world}: $World\ Gaze\ Position$ X
4. z_{world}: $World\ Gaze\ Position$ Z
5. ts_{gaze} : $Timestamp\ of\ Gaze\ Event$

A series of functions are used in the transformation of the original variables into derived measures to characterize, understand, and predict attention patterns from 3D eye movements. Vectors in each of the three directions are defined as starting at the head position and reaching towards the world gaze position, as follows:

$$\begin{cases} \widehat{x_o} = x_{world} - x_{head} \\ \widehat{z_o} = z_{world} - z_{head} \end{cases} \quad (1)$$

The norm of each vector is 1, as they are all unit vectors.

$$\|\hat{x}\| = \|\hat{y}\| = \|\hat{z}\| = 1 \quad (2)$$

Vectors are normalized into unit vectors.

$$\|\widehat{xz}\| = \sqrt{\widehat{x_o}^2 + \widehat{z_o}^2} \quad (3)$$

Below, the normalized gaze vectors in the X and Z directions are obtained by calculating the square root of the sum of the squares of the differences between world gaze coordinates and head position coordinate in each respective direction. The formula below depicts this relationship:

$$X_{3Dgaze} = \frac{\widehat{x_o}}{\|\widehat{xz}\|} = \frac{\widehat{x_o}}{\sqrt{\widehat{x_o}^2 + \widehat{z_o}^2}} \quad (4)$$

$$Z_{3Dgaze} = \frac{\widehat{z_o}}{\|\widehat{xz}\|} = \frac{\widehat{z_o}}{\sqrt{\widehat{x_o}^2 + \widehat{z_o}^2}} \quad (5)$$

Figure 5-3 visualizes the resulting projection applied to all points in a trial. All units displayed are in meters.

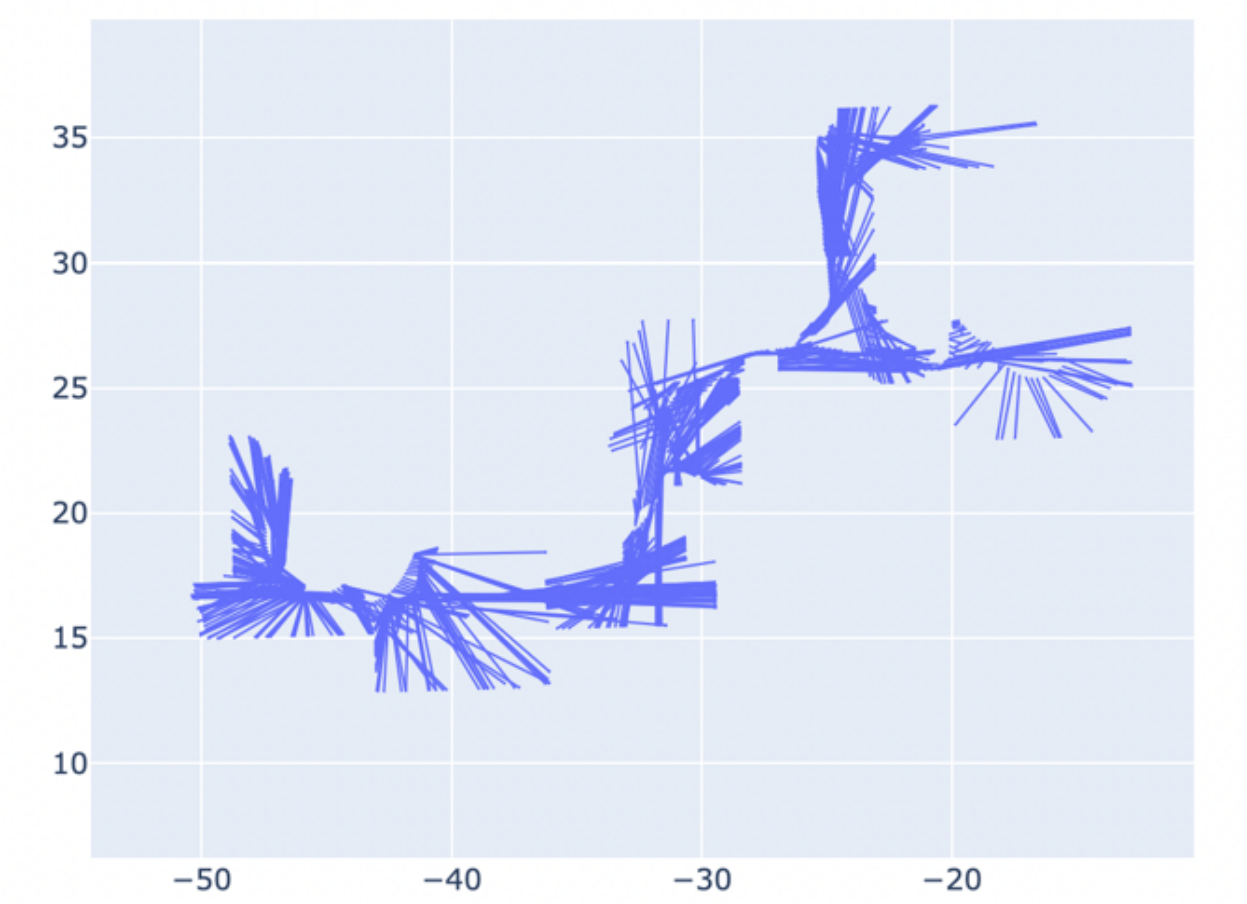


Figure 5-3. All vectors pointing from head outward towards the gaze location

5.4 Head Rotation

The angle measures in the native files include Euler and Quaternion rotation (Zaman, 2022). Euler angle is isolated for analysis. Specifically, the head rotation in the Euler Y direction is of interest. The Euler angles in the X and Z directions showed minimal variability, indicating that most of the movement happened around the Y direction – "looking around" rather than "looking up and down" or "tilting" the head by rotating the ear towards the shoulder.

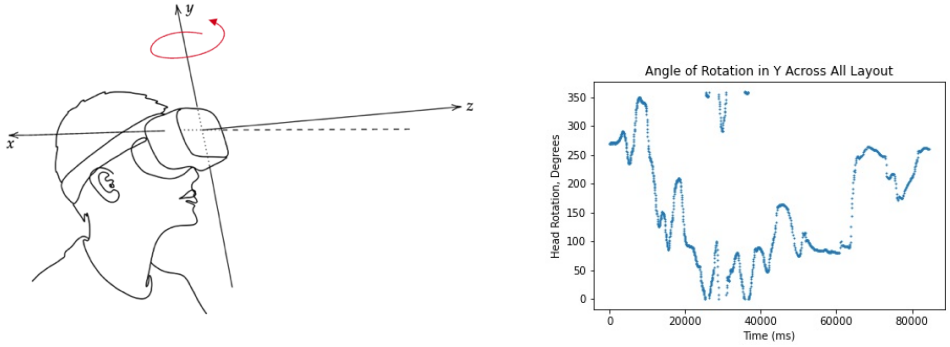


Figure 5-4. VR 3D Euler Rotation in Y Dimension

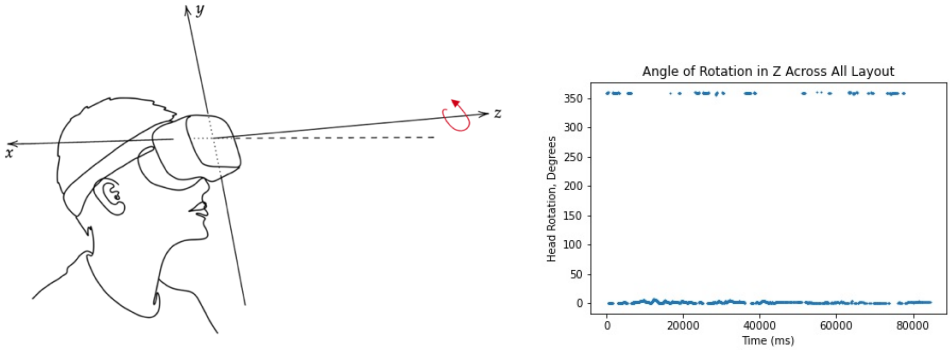


Figure 5-5. VR 3D Euler Rotation in Z Dimension

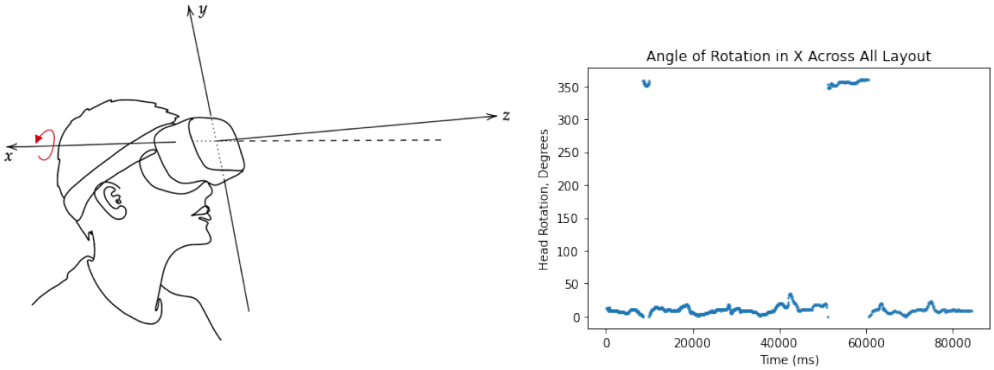


Figure 5-6. VR 3D Euler Rotation in X Dimension

Note: Coordinate system and direction of rotation adapted to stock image from: *Man in Future Living Metaverse* [Vectors], designed by Cinz, PNGTree.https://pngtree.com/freepng/man-in-future-living-metaverse_6960080.html

Variation in rotation along all three dimensions in visualized in Figure 5-7. The largest variation is seen across the Y axis.

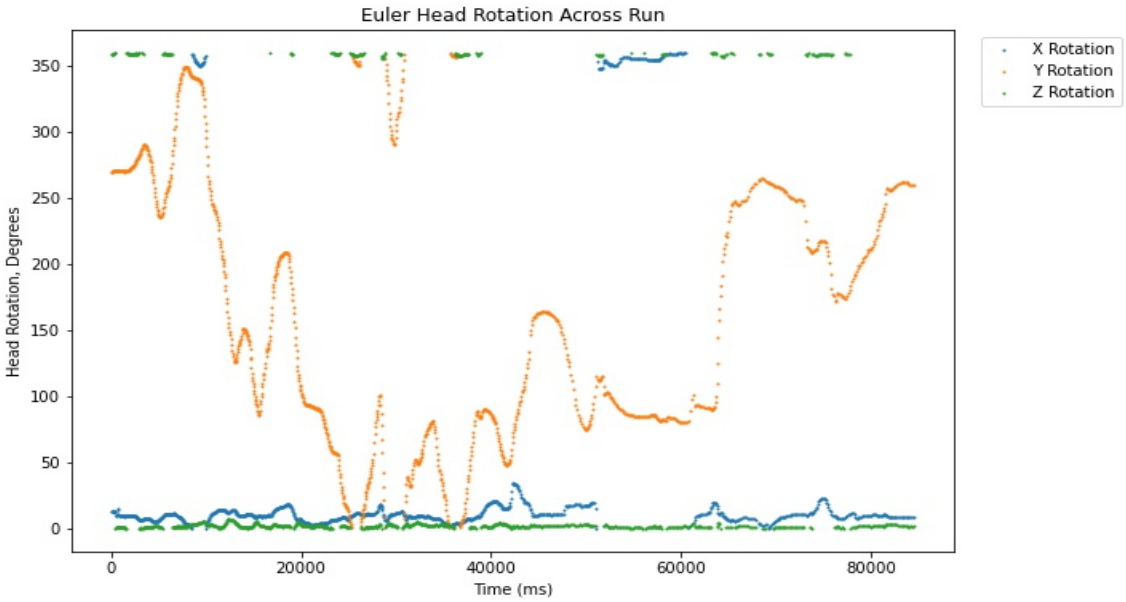


Figure 5-7. Variation in Rotation in X-Y-Z planes

The Euler angles in the X and Z directions show minimal variability, indicating that most of the movement happened around the Y direction – "looking around" rather than "looking up and down" or "tilting" the head by rotating the ear towards the shoulder.

5.5 Direction of Head, Gaze, Motion

The length **ScreenGazeZ** length is extended along the orientation of the Euler Head Rotation to obtain the vector in the direction of Head Orientation. First, a function is set up to convert degrees to radians.

$$x_{head} = -sin\left(-\frac{\pi}{180} \times \theta_y\right)$$

$$z_{head} = -cos\left(-\frac{\pi}{180} \times \theta_z\right)$$

where:

$$\theta_y\colon\ head\ rotation\ angle\ across\ Y\ axis$$
$$\theta_z\colon\ head\ rotation\ angle\ across\ Z\ axis$$

Extension of the Screen Gaze into the Z direction results in vectors of equal length to the vectors from the head position. In Figure 5-8, the RED line depicts head orientation, the BLUE line shows the orientation of the gaze, and the GREEN line shows the direction of motion.

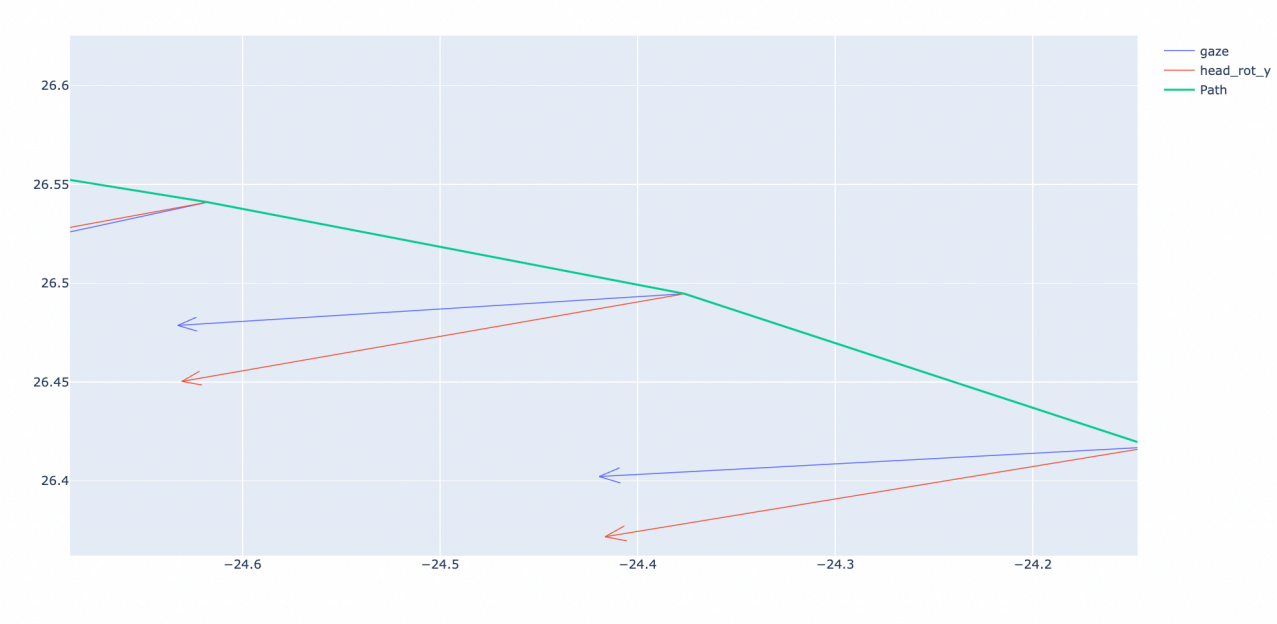
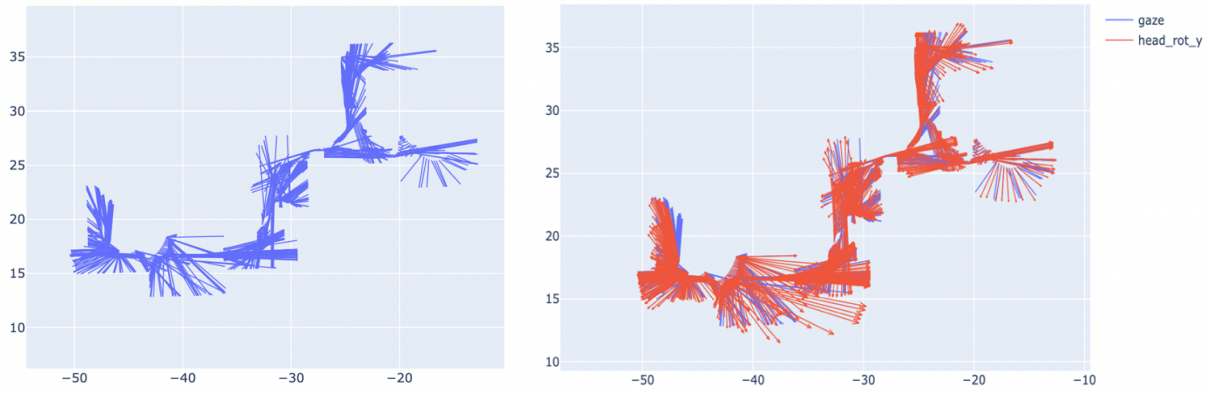


Figure 5-8. Visual angles between path, gaze vector, and head orientation.

The blue vector starts where the subject was standing and extends to the location of the world gaze. The red vector shows the direction the head was pointed, calculated by extending the ScreenGazeZ along the direction of Head Rotation (Euler).



(a) (b)

Figure 5-9. Gaze orientation vectors (a), with overlay of the new head orientation vector (b)

The path traversed by the participant is then added to the vector visualization (Figure 5-10).

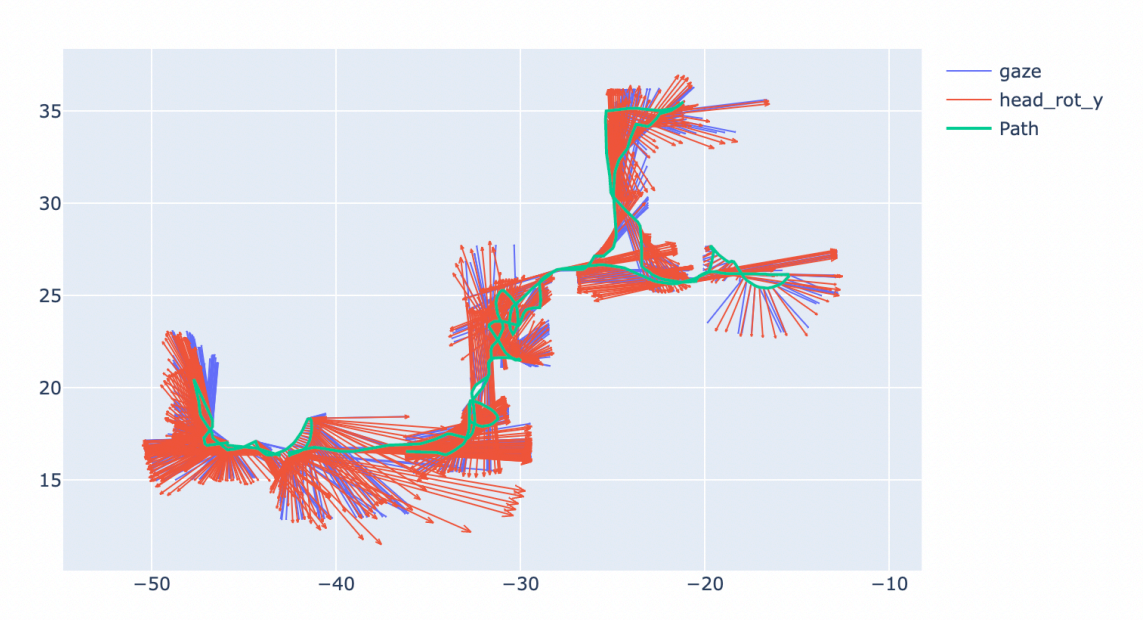


Figure 5-10. Head Vector, Gaze Vector, and Path

Vectors are scaled down to allow clear visibility onto the visual angle at each turn (Figure 5-11).

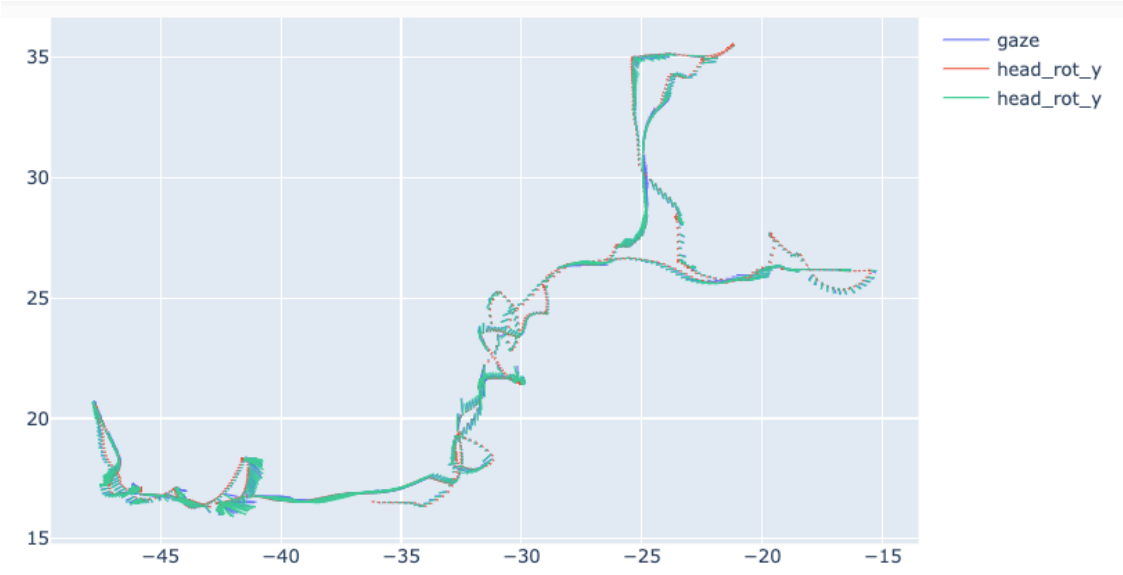


Figure 5-11. All vectors are scaled to 10% original size for angle visibility

5.6 Trigonometric Transformations

Once all vectors were determined, trigonometric calculations revealed visual angles of interest. The dot product and the cross product were both evaluated as methods to calculate the angle between the head orientation vector, gaze direction vector, and direction of walking vector.

Dot Product

$$\theta = cos^{-1}\left(\overrightarrow{u} \cdot \overrightarrow{v}\right) / \left(\| \overrightarrow{u} \| \| \overrightarrow{v} \|\right) \quad (1)$$

Cross product

$$\theta = sin^{-1}\left(\overrightarrow{u}\, x\, \overrightarrow{v}\right) / \left(\| \overrightarrow{u} \| \| \overrightarrow{v} \|\right) \quad (2)$$

One methodological problem encountered is the motion crossing the "north" coordinate system, which appear like "jumps" in the continuous representation of Euler angles in a 2D scatterplot.

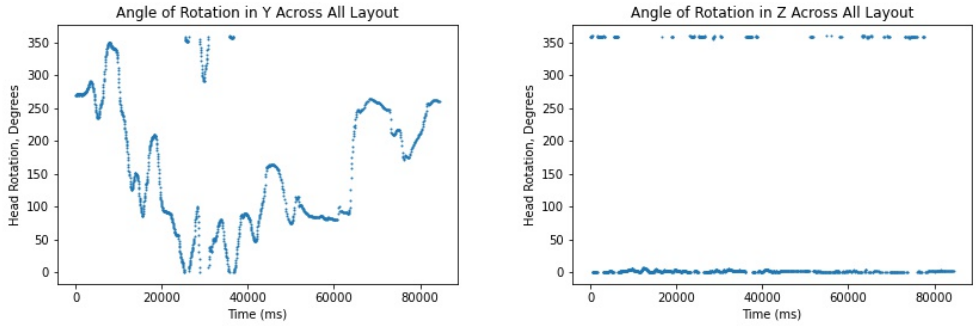


Figure 5-12. Euler Angle Jumps Across 0 Degrees

To resolve this, both sin and cosine functions were evaluated as potential functions. Sine and cosine trigonometric transformations of the angles of rotation are both visualized in Figure 5-13 to establish a viable method for extending the gaze vector into the direction of the head orientation. The unit circle with trigonometric relationships is provided in figure 5-14 for reference.

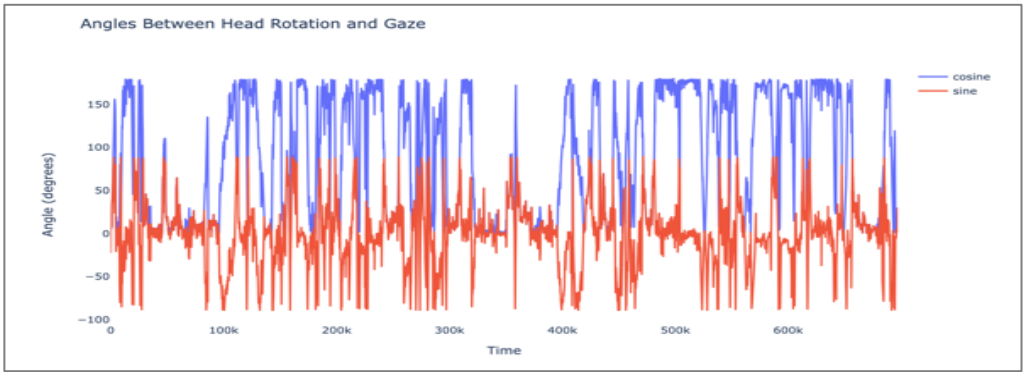


Figure 5-13. Cosine and Sine Function Evaluation

The sign of the Sine function (red) stays consistent with the sign of the angles in our range (-90º to +90º). In other words, when the angle is positive, the sign of the angle is positive, and when the angle is negative, the sign of the angle stays negative, allowing derived vectors to reflect the true direction of the movement. The sine of the angle was ultimately selected for sign consistency.

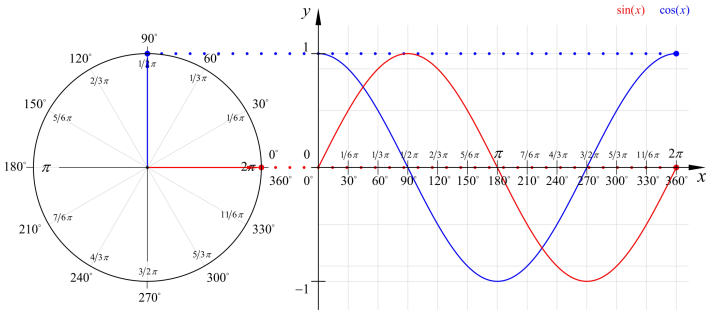


Figure 5-14**.** Unit Circle Trigonometric Relationship Reference

Note. From *Education Series Math Sine Cosine Waveforms and Phasor Diagram 360 Degrees* [Images], by: 3P-Voltage, 2024 AdobeStock. https://stock.adobe.com/images/education-series-math-sine-cosine-vector-line-diagram-360d-degrees/249875760

Figure 5-15 shows the result of the cleaning function across jumps: before and after.

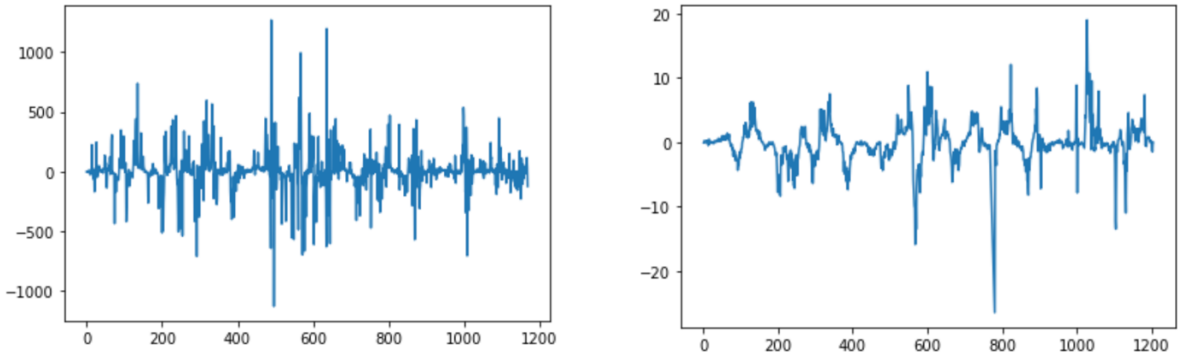


Figure 5-15. Change in speed before cleaning angles (a) and after cleaning protocol (b)

5.7 Angular Velocities

Figure 5-16 shows the ranges in rotation speed (in degrees per second) for between the gaze vector and head orientation (blue) and between the walking direction and head orientation.

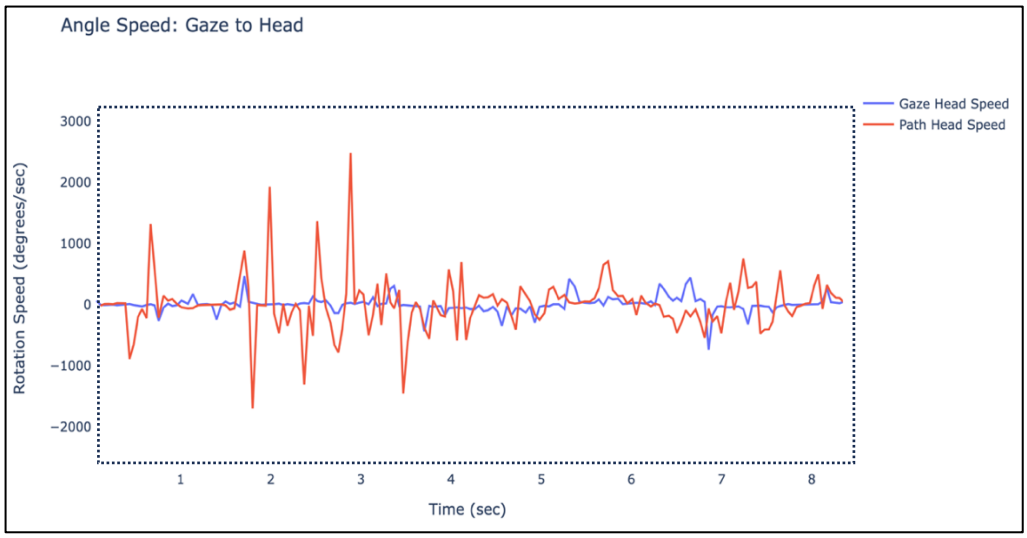


Figure 5-16. Angular velocities, gaze-to-head (blue) and head-to-path (red)

An interesting observation is that the ranges in rotation speed between the head and walking direction (path) spike to 2000 degrees in both positive and negative direction. These are not motions which are natural to human motion and point to an artefact of the joystick-driven movement in the Virtual Reality headset. This feature is removed from subsequent analyses as readings point to an equipment error.

Once all 3D features were derived, all points were visualized. A visualization tool of choice for this exercise is VisIt[1] (Childs, 2012), an open-source interactive visualization/ animation tool by Lawrence Livermore National Labs[2] (2022). Figure 5-17 shows an example view into all 1.34 million head positions (color gradient) and gaze coordinates (black) measured for all 48 participants across all experiments. Chapter 7.2 further details specific observations from these visualizations.

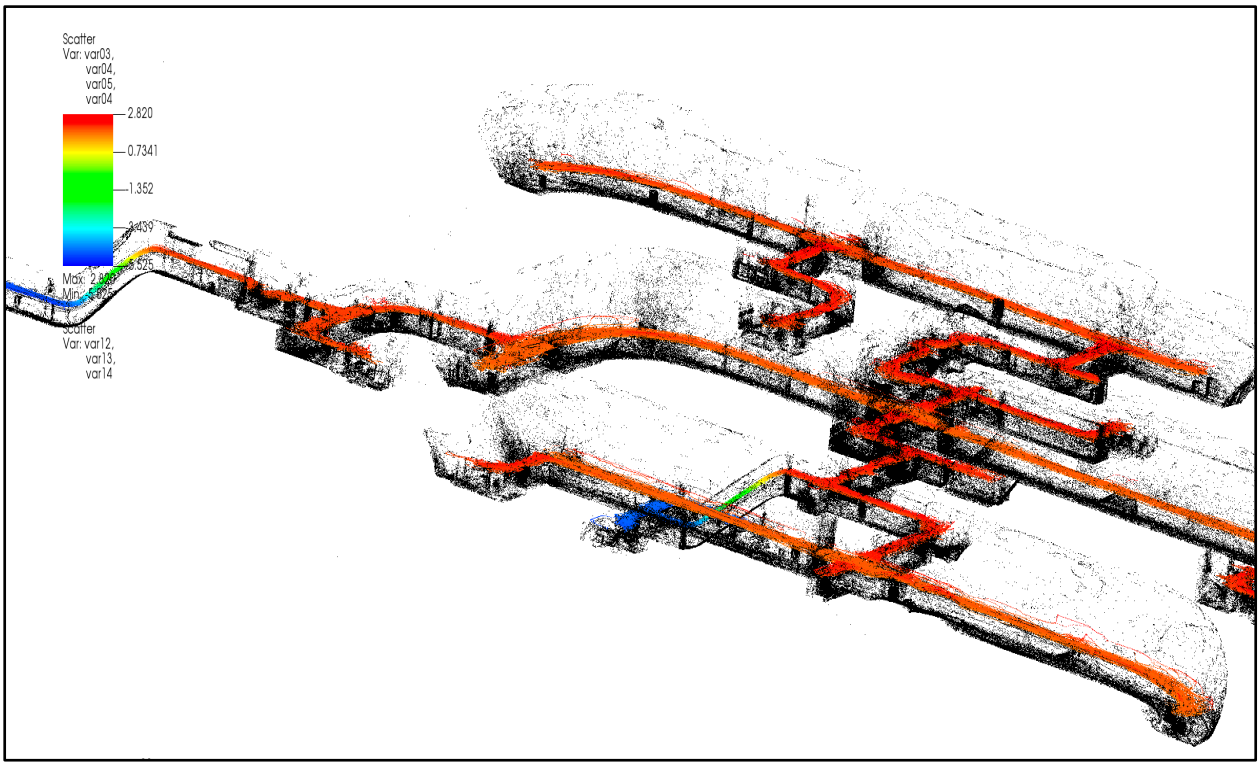


Figure 5-17. All Gaze Coordinates in VisIt 3D Visualization Tool

[1] Accessed (2023) from: https://visit-dav.github.io/visit-website/.
[2] Accessed (2023) from: https://sd.llnl.gov/simulation/computer-codes/visit

CHAPTER 06:

2D RESULTS

6 2D RESULTS

The following chapter details the results of all studies performed in the 2D Space. Each topic is organized by target variable, with subchapters indicating the independent variables that are regressed onto the target. Analyses are organized in increasing complexity, presenting first supervised learning models trained on one preliminary subject, followed by supervised models trained on all subject data, and concluding with the results of unsupervised models.

- Response Time
  - Distance vs. Response Time
    - Linear Regression Analysis: Preliminary 1-Subject Study
    - Linear Regression Analysis: 20-Subject Study
  - Gaze Path Length vs. Response Time
    - Linear Regression Analysis: Preliminary 1-Subject Study
    - Linear Regression Analysis: 20-Subject Study
    - Logistic Regression Analysis: Preliminary 1-Subject Study
    - Multiple Logistic Regression Analysis: Preliminary 1-Subject Study
    - Decision Tree: Preliminary 1-Subject Study
  - All Features vs. Response Time:
    - Logistic Regression: 48-Subject Study
    - Decision Tree: 48-Subject Study
- Attentional Shift
- Light Status Study

RESPONSE TIME / DISTANCE STUDY

6.1 Response Time Analysis

6.1.1 Response Time vs. Distance

Question 1.1: To what extent is there a correlation between 2D gaze-to-stimulus distance and task response latency?

H_1(o): No correlation exists between gaze-to-stimulus 2D geometric distance and response time.

H_1(a): If the stimulus location lies further in the field of view from the visual center of the gaze, the time it takes to respond to the stimulus is longer.

Figure 6-1 depicts all instances of correct secondary light tasks events isolated for one participant navigating all four environments. The average distance between the gaze center and the light stimulus is visualized for every isolated event.

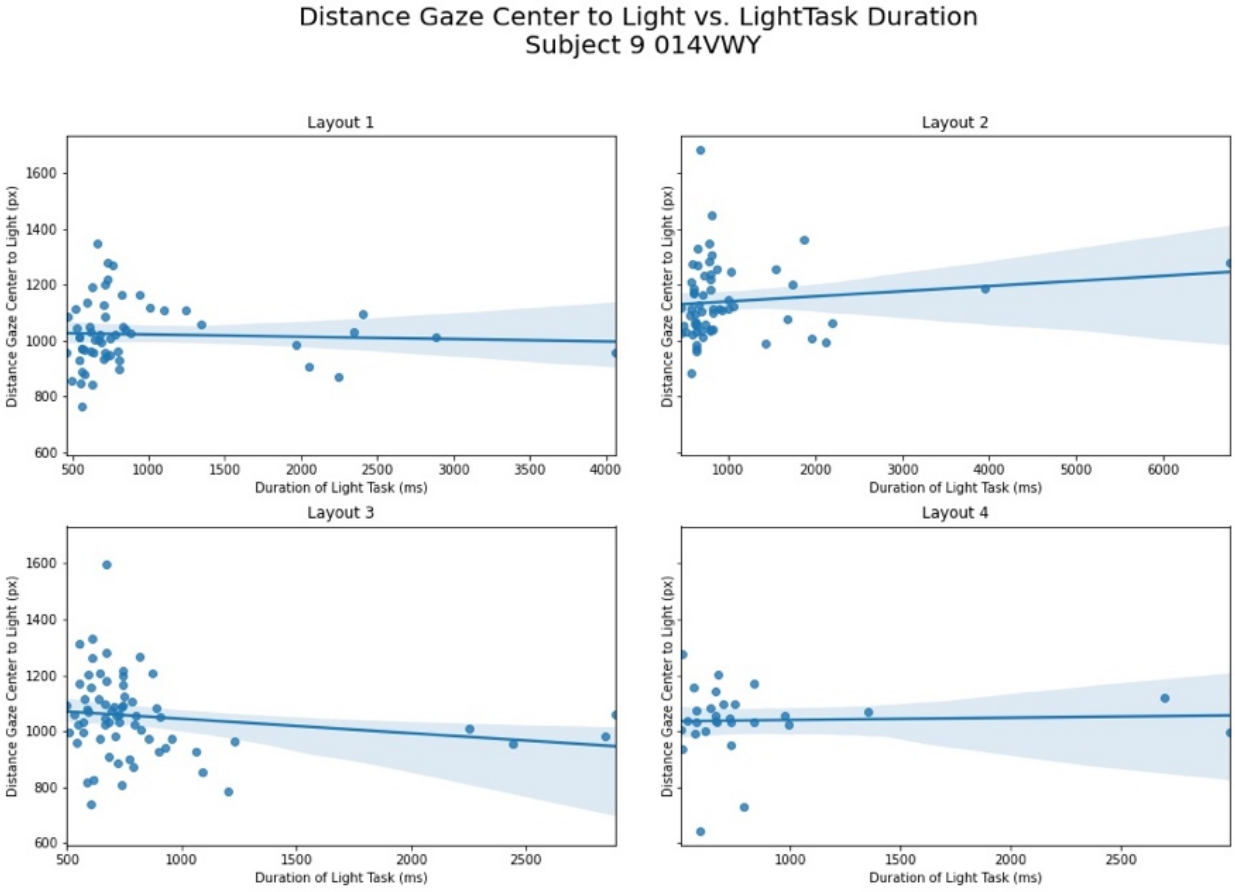


Figure 6-1. Gaze Distance to light vs. Light Task Response Time.

To establish a baseline correlation, a simple linear regression is fitted to the data to understand the amount of variation in the reaction time that is attributable to the gaze-to-stimulus distance. For the preliminary one-subject analysis, R^2 value of 0.001 for the linear regression of all four trials indicates that there is no correlation between the average gaze-stimulus distance and the reaction time to complete the secondary task.

Table 17. Linear Regression, Distance and Response Time, 20 subjects

	Distance: Linear Regression R^2									
	S01	**S02**	**S03**	**S04**	**S05**	**S06**	**S07**	**S08**	**S09**	**S10**
Layout 1	0.0014	0.0332	0.0034	0.0163	0.0005	0.0005	0.0104	0.0007	0.0024	0.0023
Layout 2	0.0122	0.0012	0.0028	0.0013	0.0138	0.0138	0.0104	0.0008	0.0166	0.0013
Layout 3	0.0010	0.0068	0.0013	0.0171	0.0024	0.0024	0.0195	0.0002	0.0296	0.0166
Layout 4	0.0010	0.0002	0.0082	0.0660	0.0030	0.0030	0.0007	0.0050	0.0015	0.0058
Average	**0.0039**	**0.0103**	**0.0039**	**0.0252**	**0.0049**	**0.0049**	**0.0102**	**0.0017**	**0.0125**	**0.0065**
	Distance: Linear Regression R^2									
	S11	**S12**	**S13**	**S14**	**S15**	**S16**	**S17**	**S18**	**S19**	**S20**
Layout 1	0.0005	0.0007	0.0009	0.0000	0.0125	0.0138	0.0056	0.0277	0.0000	0.0043
Layout 2	0.0425	0.0006	0.0532	0.0209	0.0005	0.0087	0.0004	0.0066	0.0026	0.0037
Layout 3	0.0094	0.0023	0.0026	0.0195	0.0014	0.0018	0.0323	0.0272	0.0001	0.0001
Layout 4	0.0004	0.0064	0.0476	0.0000	0.0071	0.0281	0.0125	0.0159	0.0016	0.0554
Average	**0.0132**	**0.0025**	**0.0261**	**0.0101**	**0.0054**	**0.0131**	**0.0127**	**0.0193**	**0.0011**	**0.0158**

No signal is identified, indicating that direct perception alone does not explain the mechanism by which the participant notices and extinguishes the light. The null hypothesis is maintained. The absence of a correlation prompts further investigation into the remaining 2D visual attention features to understand their contribution to the mechanism of visual perception.

6.1.2 Response Time vs. Path Length Travelled

Question 1.2: To what extent is there a correlation between task response time and gaze path length travelled over the course of the task?

H_1(o): No correlation exists between gaze path length and task response time.

H_1(a): As the response time to the stimulus increases, the gaze path travel length increases.

Experimental Design 1.2

Question 1.2 Linear Regression Results, One Subject (52 Events)

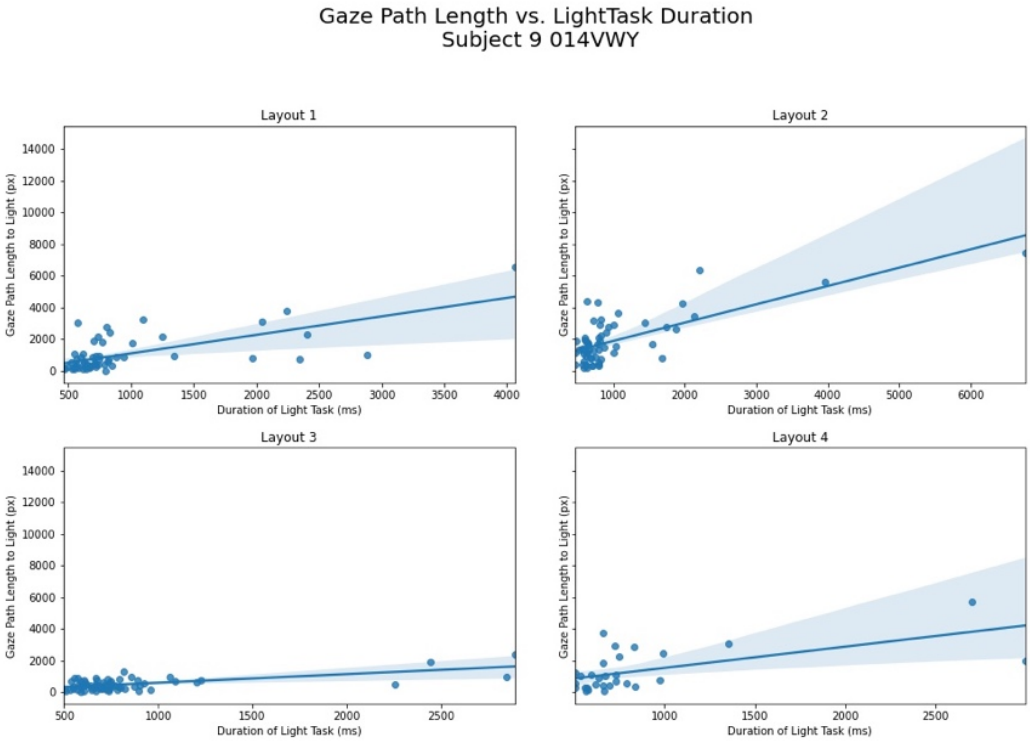


Figure 6-2. XY Path Length (px) vs. Time to Extinguish Light (ms)

A positive correlation is identified across the 52 events isolated for one subject (**R^2 = 0.78**).

Table 18. Linear regression, Gaze path and response time, 20 subject

	Gaze Path: Linear Regression R^2									
	S01	**S02**	**S03**	**S04**	**S05**	**S06**	**S07**	**S08**	**S09**	**S10**
Layout 1	0.77886	0.9834094	0.69225	0.927239	0.565372	0.565372	0.787968	0.9876	0.4458454	0.8201105
Layout 2	0.87408	0.9185376	0.52999	0.994802	0.399666	0.399666	0.486654	0.33075	0.5177371	0.7861192
Layout 3	0.87616	0.8310539	0.77451	0.948882	0.615869	0.615869	0.150674	0.68995	0.4131146	0.826846
Layout 4	0.76804	0.8930659	0.76859	0.966794	0.199646	0.199646	0.417047	0.25636	0.348085	0.6258028
Average	**0.8243**	**0.9065**	**0.6913**	**0.9594**	**0.4451**	**0.4451**	**0.4606**	**0.5662**	**0.4312**	**0.7647**

	Gaze Path: Linear Regression R^2									
	S11	**S12**	**S13**	**S14**	**S15**	**S16**	**S17**	**S18**	**S19**	**S20**
Layout 1	0.72274	0.5473226	0.6013	0.168577	0.932645	0.668159	0.385753	0.01597	0.6684519	0.9662852
Layout 2	0.65327	0.7735613	0.84152	0.011322	0.911717	0.664776	0.722165	0.81295	0.7599741	0.6541952
Layout 3	0.71701	0.4136211	0.51393	0.07702	0.496737	0.827509	0.778763	0.46974	0.685302	0.9825411
Layout 4	0.40265	0.3385444	0.67594	0.369727	0.695846	0.656446	0.862942	0.44949	0.9319622	0.7032515
Average	**0.6239**	**0.5183**	**0.6582**	**0.1567**	**0.7592**	**0.7042**	**0.6874**	**0.4370**	**0.7614**	**0.8266**

The positive correlation is maintained when observed across 20 subjects, with linear correlation coefficients ranging from 0.15 to 0.90 and averaging at 0.63 across all 4 layouts for all 20 subjects.

The null hypothesis is rejected, as the trends support the hypothesis that gaze path travelled increases with response time. However, it is important to note that this study was limited to a reduced sample size (first 20 subjects) as this study was performed earlier in this work using a manual and time-intensive process. Considerations for future work include removing outliers beyond the 2000ms timestamp, as visual interpretation of the regression results suggests that shorter response times may yield a different correlation factor.

6.1.3 Preliminary Study: Logistic Regression Response Time Results

Question 1.2 Single-Variable Logistic Regression Results, One Subject (52 Events)

Preliminary logistic regression models were trained on subject data to understand the individual contribution of input features independently towards the response time target variable according to the experimental design. The models were evaluated for consideration in the comprehensive all-subject study using *prediction accuracy* as the performance metric. One *Eye-Tracking file* and the associated *Light-Task file* for one subject were processed using algorithm xx, resulting in two datasets formatted for supervised machine learning analysis: (1) the cleaned and *Annotated Gaze Dataset* and (2) the *Events-Level Summary Statistics Dataset*.

Model input features were selected individually to train single-variable logistic machine learning models. The results of five distinct single-variable logistic regression models fitted to this data are listed in Table 19 and models are visualized in subsequent figures.

Table 19. Response Time Logistic Regression: Single Variable, Single Subject

Target	Classes	Threshold (ms)	Model Input Features	Prediction Accuracy	*Logistic Intercept*	*Logistic Coefficient*
Response Time	1. Below threshold 2. Above threshold	2000	1. Total Path Length	**0.875**	-3.46	0.0013
		1000	2. Total Path Length	**0.938**	-3.82	0.0026
		1000	3. Gaze Dispersion Y	0.688	-1.73	0.0275
		1000	4. Gaze Dispersion X	**0.75**	-1.32	0.0125
		1000	5. Distance to Light	0.438	-3.5×10^{-9}	3.9×10^{-6}
					Dataset: Biglights1	

The resulting logistic regression curves presented in Figures 6-3 show a graphical representation of how changes in each model input feature (shown on the x-axis) correspond to the probability of the target variable class.

The first two models establish a baseline relationship between the Response Time Class (Below Threshold) and (Above threshold) and the Total Path Length. The two models defined use different thresholds to distinguish the upper and lower classes: 1000ms (Figure 6-3a) and 2000ms (Figure 6-3b). The model equations below represent each respective relationship fitted to the data:

$$z(RT)_{t=1000} = -3.82 + 0.0026 * L_{gpt}$$

$$z(RT)_{t=2000} = -3.46 + 0.0013 * L_{gpt}$$

where L_{gpt} represents the Length in Gaze Path across all of the gaze points pertaining to the duration of the light task, and $z(RT)_t$ represents the log-odds (logit) of the probability of the positive class: **above threshold** (where t = threshold in milliseconds). Here, the logistic coefficient of 0.0026 for the 1000ms-class threshold is larger than the logistic coefficient calculated for the 2000ms-split classes. Both coefficients are positive, indicating positive relationships in both cases. The larger coefficient for the RT-class split at t=1000ms reflects a stronger relationship between the Total Path Length and Response Time Class split at 1000ms over the 2000ms-split. The prediction accuracy of 0.938 for $(RT)_{t=1000}$ is higher than the accuracy of 0.875 for the model fitted to target $(RT)_{t=2000}$.

```
Logistic Model Intercept: [-3.82096752]
Logistic Model Coefficient: [[0.00262301]]
```

```
Logistic Model Intercept: [-3.45975222]
Logistic Model Coefficient: [[0.00128286]]
```

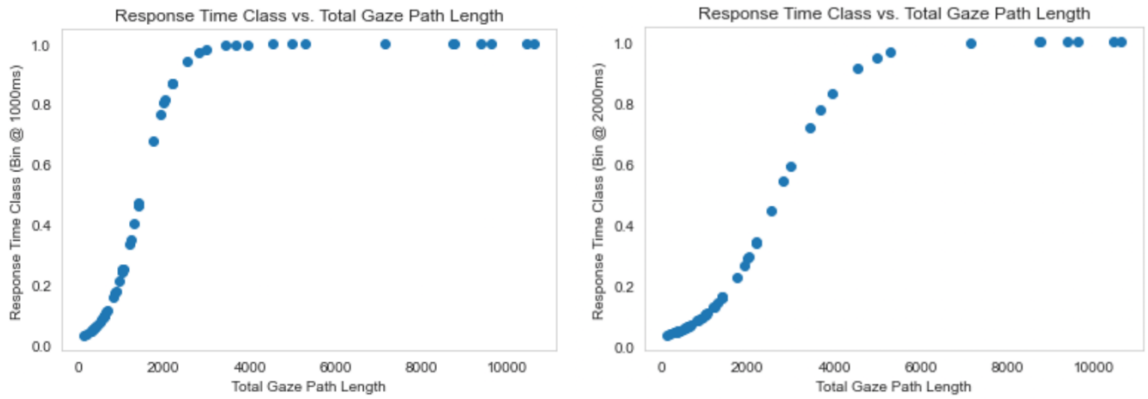


Figure 6-3. Response Time Logistic Reg:(a)1000ms (b)2000ms vs. Gaze Path Length(px)

Figure 6-3 illustrates the resulting regression of the Response Time (time to extinguish the light) onto the independent variable, Total Gaze Path Length across all light events for one subject. The S-curve shows the extent to which variations in the Total Gaze Path Length correspond to the probability of a shift in the Response Time. The magnitude of the coefficient for Model 1 is directly correlated to the slope of the logistic regression curve. Figure 6-3 shows a steeper incline for the Response Time Threshold split at t=1000ms (Figure 6-3a) than at t=1000ms (Figure 6-3b), indicating that small changes in Path Length have a more pronounced effect on the probability of the positive Response Time Class at the lower threshold.

The logistic regression can discriminate more effectively between the Total Gaze Path Length corresponding to the Response Time Class binned at 1000ms as compared to 2000ms, a finding consistent with the higher accuracy of the model. Based on this finding, the **threshold of 1000ms is selected** and held constant for all subsequent machine learning models which discern between binary classifications of Response Time as a target variable.

6.1.4 48-Subject Results: Logistic Regression

Table 12 lists the results of all simple and multiple logistic regression models trained on the Aggregate 2D Dataset, with data from all 4-layouts to all 48-subjects.

Table 12. Response Time Logistic Regression, Single and Multivariate: 48-Subject

Target	#	Classes	Input Features	Model	Accuracy
Response Time	1	1. Less than 1 sec 2. More than 1 sec	1. Distance to Light	Logistic Regression	**0.708**
	2	1. Less than 1 sec 2. More than 1 sec	1. Gaze Dispersion X	Logistic Regression	**0.709**
	3	1. Less than 1 sec 2. More than 1 sec	1. Gaze Dispersion Y	Logistic Regression	**0.726**
	4	1. Less than 1 sec 2. More than 1 sec	1. Total Gaze Path 2. Average Gaze Speed	Logistic Regression	**0.859**
		1. Less than 2 sec 2. More than 2 sec	1. Total Gaze Path 2. Average Gaze Speed	Logistic Regression	0.925
	5	1. Less than 1 sec 2. More than 1 sec	1. Distance to Light 2. Total Gaze Path 3. Average Gaze Speed 4. Gaze Dispersion X 5. Gaze Dispersion Y	Multiple Logistic Regression	0.883
	6	1. Less than 500ms 2. More than 500ms	1. Distance to Light 2. Total Gaze Path 3. Event Gaze Speed 4. Gaze Dispersion X 5. Gaze Dispersion Y	Multiple Logistic Regression	0.903

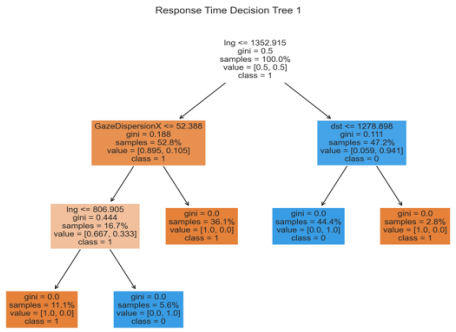
6.1.5 Response Time Multiple Feature Classification

Decision trees were trained for the preliminary dataset with all 6 input features: the distance, length, speed, position, and gaze dispersions in X and Y directions. The target variable was set as the Response Time, and target outcome classes were again set to a threshold of 1000ms, with the models aiming to predict the classification of response time into less than or greater than the threshold. Model input parameters are:

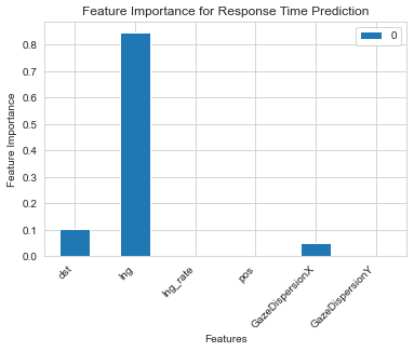
Target Variable	*Target Outcomes*	*Independent Variables*
Response Time	Less Than 1000ms	Distance
	More Than 1000ms	Length
		Speed
		Position
		Gaze Dispersion X
		Gaze Dispersion Y

Decision tree model results are presented in detail Figures 6-4 and 6-5.

The all-feature decision tree model shown in in Figure 6-4 predicted response time with 87.5% accuracy. The feature with the highest contribution to the prediction accuracy was the gaze path length travelled.



(a) tree split visualization



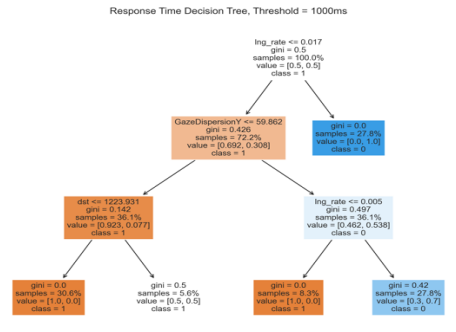
(b) feature importance

Feature	%
dst	0.10
lng	**0.84**
lng_rate	0.00
pos	0.00
GazeDispersionX	0.05
GazeDispersionY	0.00
Performance	
Accuracy:	**0.875**
Precision	1.0
Recall:	0.75

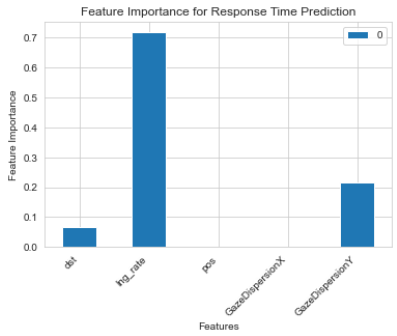
(c) performance

Figure 6-4 Response Time Decision Tree: (a) splits, (b) feature importanc, (c) performance

Figure 6-5 shows the decision tree classification paths visualized in the absence of the most prominent feature. Among remining features, the highest contributing visual attention attribute to the response time is the gaze speed (lng_rate), and the accuracy of the model is 81.3%.



(a) tree split visualization



(b) feature importance

Performance	
Accuracy:	**0.813**
Precision	0.778
Recall**:**	0.875
Feature	
dst	0.07
lng_rate	0.72
pos	0.00
GazeDispersionX	0.00
GazeDispersionY	0.2

(c) performance

Figure 6-5. Response Time Decision Tree: (a) splits, (b) feature importance, (c) performance

The D-Tree viz algorithm (Parr, 2021) was used to look deeper into the decision tree prediction pathways for response time. The resulting histograms in Figure 6-6 show the distribution of data for each cluster at all nodes and leaves.

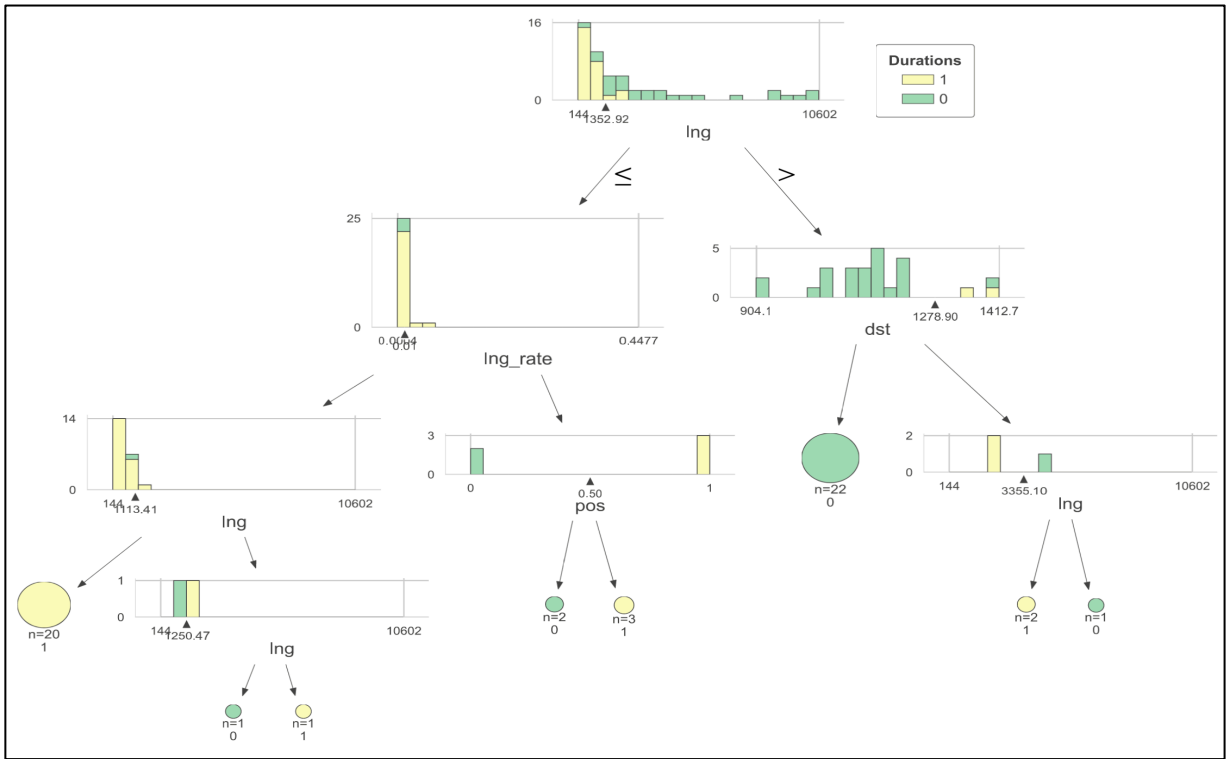


Figure 6-6. Decision Node/Leaf Histogram Tree

Note: Algorithm Copyright (c) 2021 Terence Parr. https://github.com/parrt/dtreeviz

6.1.5.1 Hyperparameter Tuning

Hyperparameter tuning involved adjusting the decision tree algorithm parameters. Tuning these specialized parameters includes setting a maximum depth for the tree, setting minimum number of samples in a leaf, and pruning the tree to minimize cost complexity and prevent overfitting.

Table 20. Predicting Response Time, Decision Tree Results

Target	Model Input Features	Run	Decision Tree		
			Accuracy	Precision	Recall
Response Time, 1s	1. Distance 2. Length 3. Speed 4. Movement Direction 5. Gaze Dispersion X 6. Gaze Dispersion Y	1	0.813	0.75	0.86
		2	0.813	0.833	0.714
	hyperparameter tuning	3	0.875	0.857	0.857

Hyperparameter tuning increased the accuracy of the decision tree model from 81.3% to 87.5% accuracy. Based on these results, the high model accuracy indicates that the decision tree can successfully discern between two response time classes based on the six input features defined.

6.1.6 Saccade/ Fixation KNN

Next, we classify gaze behaviors according to two velocity classes: Fixations and Saccades. Fixation is a type of ocular movement by which the gaze is stable and focused on one specific point within the visual field. Saccadic movement refers to a rapid eye movement by which the gaze is shifted from one point to another. Here, ***gaze speed*** is divided into two distinct categories, or behaviors, based on a threshold of 0.3 pixels per millisecond. Ocular movements with a speed below the 0.3 threshold are considered Fixations, while ocular movements with speeds above the 0.3 threshold are classified as Saccades. Binary classification is a simple initial heuristic for differentiating the two ocular behaviors and setting up preliminary machine-learning models.

Table 21. ML Predicting Behavior: Annotated Gaze Set, 1 Subject

	Target	Classes	Model Input Features	KNN Accuracy
1	**Behavior**	1. Fixation 2. Saccade	1. Speed 2. Screen Position X 3. Screen Position Y 4. Light Status 5. Distance Travel	0.908

One subject dataset (~5000 gaze movements) was used to train the k-Nearest Neighbors (KNN) algorithm to predict ocular behaviors in the 2-Dimentional Screen Space based on five input features: the Gaze Speed, the 2D Screen Position (X and Y), Light Status (On/Off) and the distanced travelled by the eye. The result (Table 21) was a high accuracy prediction of 90.8%, indicating that the KNN model was effective in distinguishing and classifying on-Screen Fixations and Saccades in various conditions and locations. This result positions the KNN as a good candidate for informing human eye interaction with on-screen AR elements in a VR simulator or in the wild using mixed-reality wearable glasses. Here, the model performed well for one participant only, so it is advanced to the all-subject analysis in Chapter 6.4.

ATTENTIONAL SHIFT STUDY

6.2 Attentional Shift Analysis

Question 5: To what extent is movement towards the location of the cognitive distraction stimulus attributed to factors other than the baseline distance?

First, a binary logistic regression was set up. The categorical response has only two possible outcomes, which state the direction of the gaze path over the course of one light task event

1. The gaze moved away from the light task element.
2. The gaze moved towards the light task element

6.2.1 Movement Direction

Dependent Variable	**Target Outcomes**	**Independent Features**	**Dataset**
Movement Direction	moved away from the light	Speed ,	light
	moved towards the light	Total Path Travelled	events

Figure 6-7 shows the change in the position at the start of the light event to the position at the end of the event. Negative differences (left) indicate all light events during which the gaze scanned towards the light. Positive differences (right) indicate all light events during which the gaze scanned away from the light. The relative duration of each event is demarcated using a color gradient (legend), with longer time events labelled as dark purple.

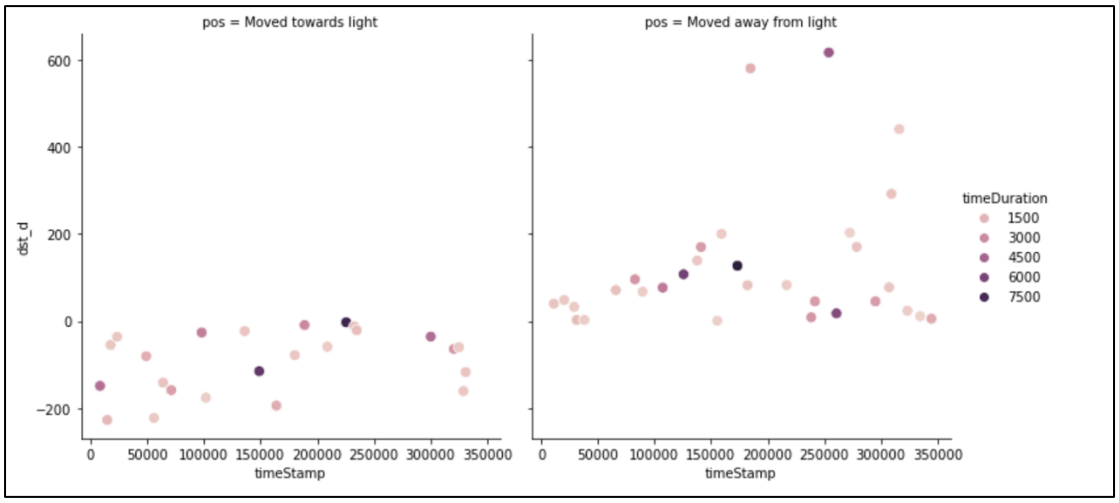


Figure 6-7. Light events: gaze scan relative to stimulus

The results of the single-variable logistic regression are below. The logistic regression model did not accurately predict the relationship between Speed or Path Length and the Movement Direction.

Target	*Classes*	*Model Input Features*	*Logistic Regression Accuracy*
Movement Direction	1. Moved Away from Light 2. Moved Towards Light	1. Speed	0.562
		1. Total Path Length	0.562

The logistic model did not accurately predict movement behavior relative to the stimulus from the total gaze speed and gaze path length across light events.

6.2.2 Movement Direction Classification, Decision Tree

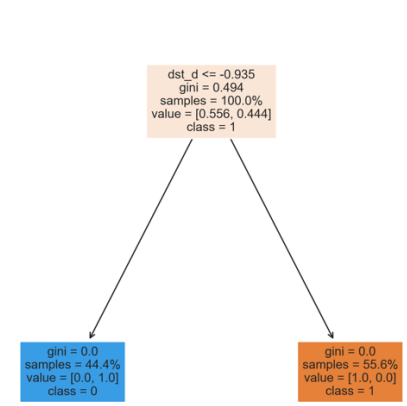
Next, several Decision Tree models were trained based on the one-subject Lights Events dataset in order to evaluate the model's ability to classify attentional states. The target variable of Movement Direction Relative to Stimulus over the course of all light event sequence is predicted from a combination of features. Several key parameters were defined to control the structure and behavior of the model. The complexity of the tree is minimized by setting the cost-complexity parameter to 0.0. The Gini impurity criterion is set to evaluate the quality of the split. Each terminal node is set to contain at least one data point, and a node is set to split only if it contains a minimum of 2 datapoints. Lastly, the maximum depth of each tree is limited to 4 levels in order to prevent excessive complexity.

Table 22. Attention Shift Decision Tree Model Input

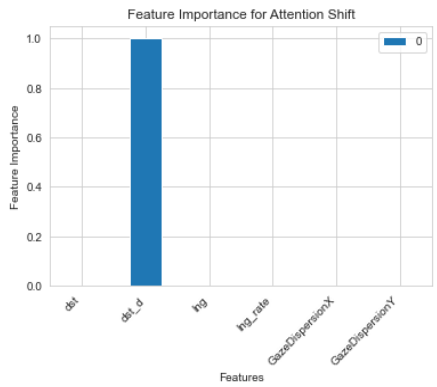
Target Variable	*Target Outcomes*	*Independent Features, Model 1*	*Independent Features, Model 2*
Movement Direction	moved away from stimulus	1. Distance 2. Change in Distance 3. Length	1. Distance 2. Length 3. Speed
	moved towards stimulus	4. Speed 5. Dispersion X 6. Dispersion Y	4. Dispersion X 5. Dispersion Y
			Data: 1 Subject, Events

A decision tree model is trained based on the one-subject light events dataset, with the target variable set to the movement direction relative to the stimulus light. The light events in the dataset were split into two classes based on the calculated movement direction: (1) movement towards the stimulus and (2) movement away from the stimulus. Two models are trained on the one-subject, 52-light event dataset.

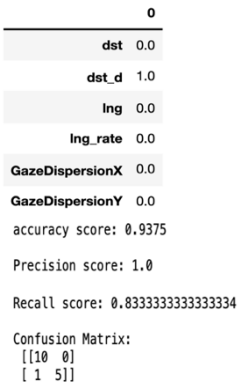
Decision Tree Model 1 utilizes the full set of independent features, while Decision Tree Model 2 excludes the feature with the highest importance. This investigation tests the hypothesis that the decision tree can accurately predict Attentional Shift over the course of a light-task event. Because the change in distance feature was used to define the movement direction, the first model is expected to perform with 100% accuracy. The importance of the remaining features on the movement direction are then investigated following the removal of the Change in Distance feature. The model results are presented and compared in Figures 6-8 and 6-9 with three key elements: (a) the decision tree visualization illustrating the sample distributions and Gini values at each leaf-node split, (b) bar charts depicting the relative importance of the independent features for each model, and (c) the model performance results, including the relative feature importance.



(a) Decision Tree 1 Visualization



(b) Feature Importance



	0
dst	0.0
dst_d	1.0
lng	0.0
lng_rate	0.0
GazeDispersionX	0.0
GazeDispersionY	0.0

accuracy score: 0.9375

Precision score: 1.0

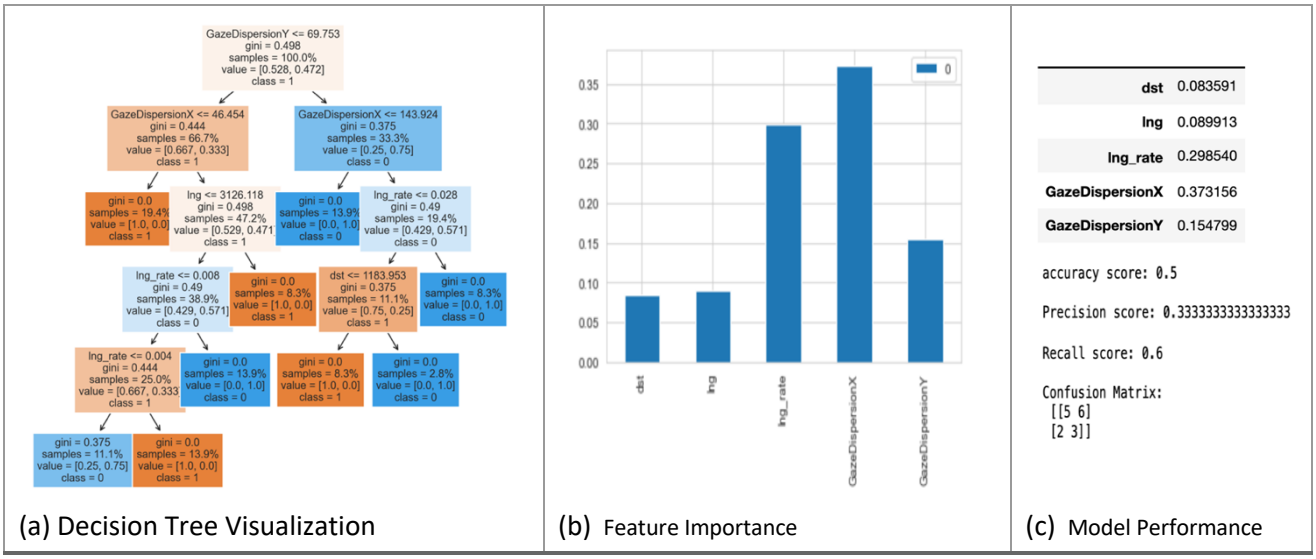
Recall score: 0.8333333333333334

Confusion Matrix:
[[10 0]
[1 5]]

(c) Model Performance

Figure 6-8. Attention Shift Decision Tree 1: (a) splits, (b) importance (c) performance

Model 1 correctly identified that Movement Direction is calculated from the change in distance across the event, more specifically, the difference between the starting and ending position across the gaze path. The decision tree visualization in Figure 6-8a shows that the model achieved an accuracy score of 93.7% in one split, with 100% of the feature importance attributed to the Change in Distance (*dst_d*) feature.



(a) Decision Tree Visualization (b) Feature Importance (c) Model Performance

Figure 6-9. Attention Shift Decision Tree 2: (a) splits, (b) importance (c) performance

Model 2 shows the result of training the model with this principal feature removed. The model with the most important feature removed performs with 50% accuracy. However, interesting patterns are seen in the remaining features. Among remining features, the highest contributing features to the Attentional Shift are gaze speed (*lng_rate*) and Gaze Dispersion in the X direction.

Table 23. Summary of Attention Shift Decision Tree Classifiers

Target	**Model Input Features**	**Decision Tree**		
		Accuracy	**Precision**	**Recall**
Movement Direction	1. Distance 2. Distance Delta 3. Length 4. Speed 5. Dispersion X 6. Dispersion Y	0.95	1	**0.8**
	1. Distance 2. Length 3. Speed 4. Dispersion X 5. Dispersion Y	0.5	.33	**0.6**

6.2.3 D-Tree Viz Node Histograms

The D-Tree Viz algorithm (Parr, 2021) was used to generate Figure 6-10, which shows the distribution of clusters at each node.

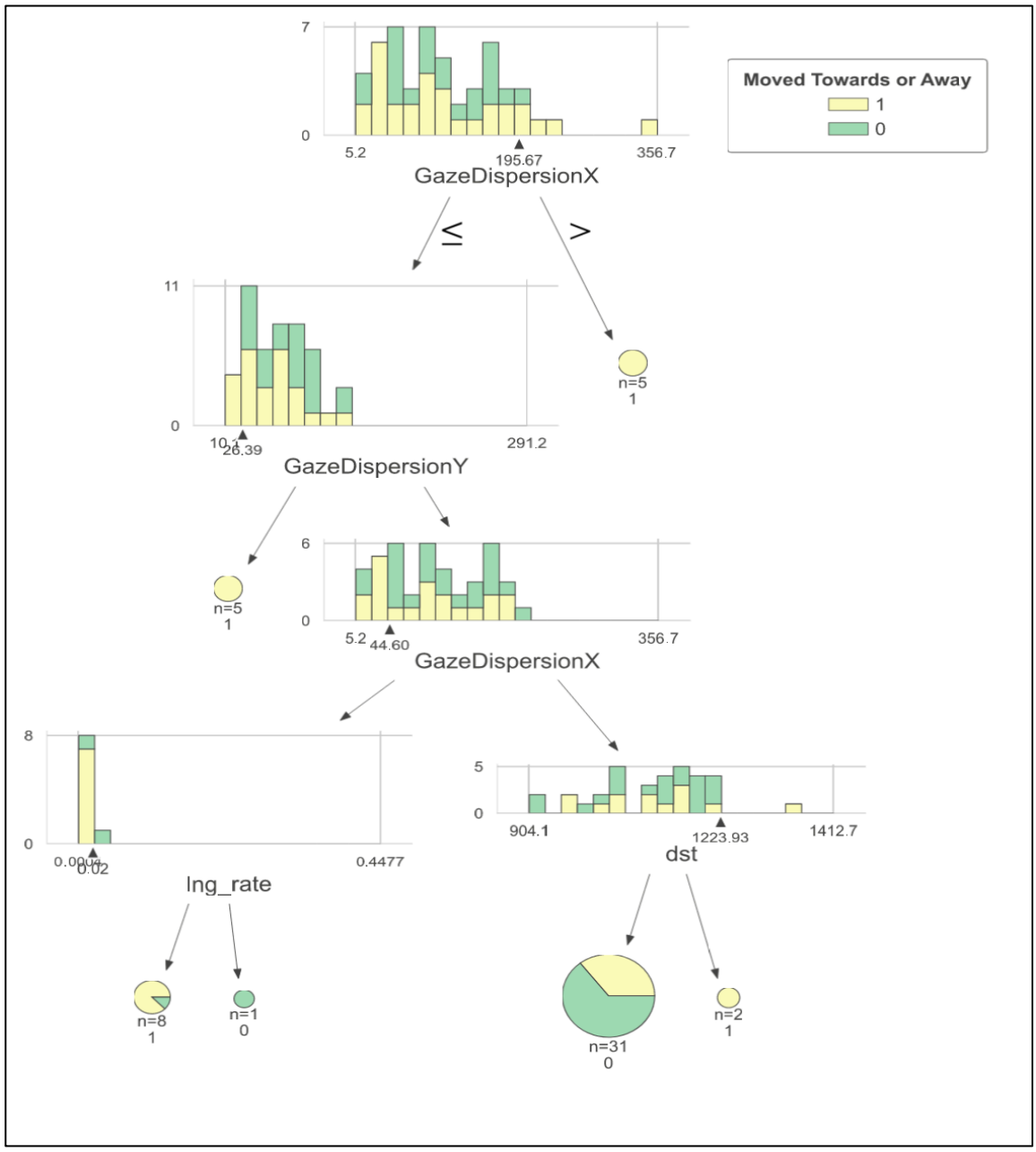


Figure 6-10. Decision Node/Leaf Histogram Tree

Note: Algorithm Copyright (c) 2021 Terence Parr. https://github.com/parrt/dtreeviz

6.2.4 ML Predicting Movement Direction: Light Events, 48 Subjects

First, a binary logistic regression was set up. The categorical response has only two possible outcomes, which state the direction of the gaze path over the course of one light task event

1. The gaze moved away from the light task element.
2. The gaze moved towards the light task element

Table 24. 2D Attention Shift 48-Subject Aggregate Regression Results

Target	*Classes*	*#*	*Input Features*	*Model*	*Accuracy*
Movement Direction	1. Moved Away from Light 2. Moved Towards Light	1	1. Total Gaze Path Travel 2. Average Gaze Speed	Logistic Regression	0.558
		2	1. Average Gaze Speed	Logistic Regression	0.558
		3	1.Gaze Dispersion X	Logistic Regression	0.558
		4	1. Gaze Dispersion Y	Logistic Regression	0.558
		5	1. Total Gaze Travel, 2. Average Gaze Speed 3. Gaze Dispersion X 4. Gaze Dispersion Y	Multiple Logistic Regression	0.558
				Total Events: 10,111 **Total Subjects:** 48 **Dataset:** 2D-ML_Lights, 48S x 4L.	

LIGHT STATUS STUDY

6.3 Light Status 48-Subject T-Test Results

6.3.1 T Test comparison of means for all Light OFF vs. Light ON Events

In the previous chapter, the relationships between two derived gaze metrics (distance to stimulus and gaze travel path) are examined for independent subjects during light events. Next, the author seeks to understand the differences between **Light-OFF** and **Light-ON** conditions across multiple features of interest. Subject data (Zaman, 2020) are compiled into one database. Algorithm 2 (python script in Chapter 3) is executed to clean and concatenate all data of interest. The original Unity Eye-Tracking files and Light Event files for all 48 subjects and all layouts are used as source data (Zaman, 2020). Data are further annotated with the associated Light Event. The resulting dataset contains **1,020,817** total gaze events in the Lights-OFF phase, and a total of **316,371** events in the Lights-ON phase across all experiments.

A two-sample T-test comparison of means for independent samples is performed on 2D features of interest: the Head Rotation in Euler Space in all 3 axes (X, Y, Z), two metrics pertaining to eye physiology (pupil diameter and eye openness averaged across both eyes), and the two previously derived gaze distance measures pertinent to 2D movement in the screen plane (gaze speed and gaze distance travelled). Results are presented in Table 25. Statistically significant differences were observed between the light ON and light OFF conditions for six out of seven gaze behaviors evaluated in the 2D screen projection (green highlight).

Table 25. T-Test differences of means between Light On/Off events for 2D features

LIGHT OFF vs. **LIGHT ON** Conditions (OFF: 1,020,817, ON: 316,371)						
	Means		**Standard Deviations**		T-Test Result	
Feature	**OFF Mean**	**ON Mean**	**OFF StdDv**	**ON StdDv**	**T Statistic**	**P Value**
Gaze Speed	1.061	1.064	1.99	2.02	-0.946	0.344
Distance Travelled	60.267	59.296	115.06	113.21	4.163	0.0000314
Head Rotation Euler X	151.675	128.930	171.77	166.20	65.572	0.0
Head Rotation Euler Y	181.854	182.485	106.63	106.39	-2.909	0.00362
Head Rotation Euler Z	111.005	123.278	162.39	167.03	-36.888	0.0
Average Pupil Diameter	5.275	5.304	1.62	1.53	-8.789	0.0
Average Openness	0.913	0.915	0.174	0.173	-3.792	0.000149

6.3.1.1 Gaze Movement Differences Between Groups

For the derived features, gaze ***Distance Travelled*** was found to be *significantly* ***lower*** in the Light-ON condition (M = 59.296, σ =113.21) than in the Light-OFF condition (M = 60.267, σ =115.06), , with a T-test statistic of 4.163 and associated P-value < 0.001. The T-test comparison of means found *no significant difference* in ***Gaze Speed*** between light conditions. The T-test statistic yielded a value of -0.946 and a corresponding p-value of 0.344 when Light-OFF group (M = 1.061, σ = 1.99) and Light-ON group (M = 1.064, σ = 2.02) were compared.

6.3.1.2 Head Rotation Differences Between Groups

Head Rotation behaviors saw significant changes across all three axes for the two groups. Head Rotation across **X-axis** was significantly **lower** in the Light-ON group (M = 128.93, σ =166.20) than in the Light-OFF group (M =151.68, σ =171.77). This feature saw the highest change across groups with a T-test statistic 65.57 and associated P-value < 0.001. Head Rotation across **Y-axis** was found to be significantly **higher** during combined Light-ON events (M = 182.49, σ =106.39) than while the Light remained OFF (M = 181.85, σ =106.63, with a T-test statistic of -2.91 and associated P-value of 0.004. Head Rotation across **Z-axis** was found to be significantly **higher** during Light-ON events (M = 123.28, σ =106.39) than while the Light remained OFF (M = 111.01, σ =162.39), with a T =-36.89 and p < 0.001.

6.3.1.3 Eye Physiology Differences Between Groups

Pupil Diameter (averaged across left and right eye) was found to be significantly **higher** in the Light-ON group (M = 5.30 σ =1.53) than in the Light-OFF group (M = 5.28, σ =1.62), where T-test statistic of – 8.79 and associated p < 0.001. **Eye Openness** (averaged across left and right eye) was found to be significantly **higher** in the Light-ON condition (M = 0.915, σ =0.173) than in the Light-OFF condition (M = 0.913, σ =0.174), where T = -3.79 and p < 0.001.

6.3.2 ML Response Time: Events KNN, 48 Subjects

The K-Nearest Neighbors model was evaluated for its ability to differentiate between 2-dimensional visual attention patterns pertaining to Light Status Classes (Light ON Events vs. Light OFF Events). Multiple kNN models were trained on a total of 10,111 light events extracted from all 48 subjects across all layouts with multiple input features.

Table 26. K-Nearest Neighbors Prediction of Light Status

Target	#	Model Input Features	KNN				
			N=50	N = 17	N = 9	N = 7	N = 5
Light Status (On/ Off)	1	1. Screen Gaze Pos X 2. Screen Gaze Pos Y 3. Change in X Position 4. Change in Y 5. Change in Time 6. Distance Travelled 7. Speed	76.31	75.8	74.32	73.41	72.03
	2	1. Speed 2. Screen Gaze Pos X 3. Screen Gaze Pos Y					71.73
	3	1. Speed					71.61
						Total Events:	10,111
						Data:	2D Lights Events, 48 subjects x 4 layouts

Table 26 shows the comparison of performances optimized across the hypothesis space with one to seven input features. The best model distinguished between the two classes: light OFF vs. light ON with N=50 neighbors and 76.31% accuracy, with the model that optimizes between computational complexity and performance identified at N=17 neighbors and 75.8% accuracy.

6.4 Unsupervised Machine Learning Clustering

6.4.1 K-Means Clustering

K means clustering was used to identify clusters in the preliminary dataset which may not be readily anticipated with manual selection of target classes. The model was trained to identify two (2) clusters from four input features:

1. event time Duration (ms)
2. Movement direction relative to light
3. Distance between Centroid and Light (px)
4. Total Travel Length (px)

K-Means Model 1

Figure 6-11 shows four scatterplots of the clustering results in 2 dimensions, where two input features are visualized at a time in each plot. Distinct clustering can be seen in the time vs. Total travel length plot (d), which is consistent with the positive relationship identified through linear regression modeling.

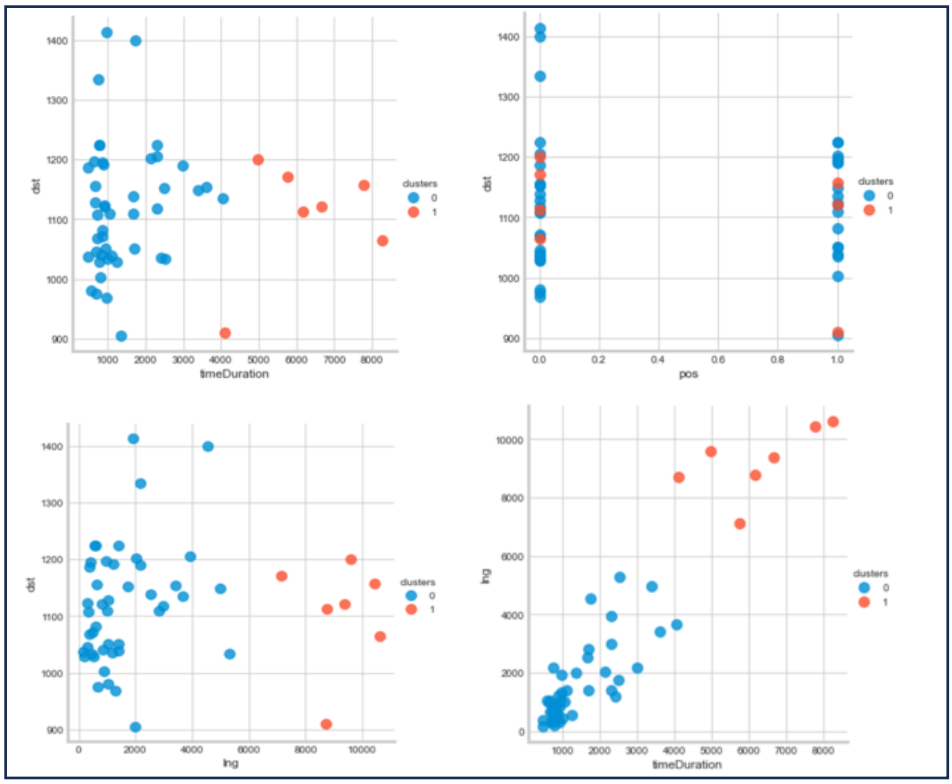


Figure 6-11. K Means Clusters of Distance and Travel Path in Light Events

K-Means Model 2

A second three-cluster k-means model was trained on the preliminary dataset of all derived features. The means of all clusters are listed in Table 27.

Table 27. K-Means 3-Cluster Parameters

clusters	timeStamp	timeDuration	dst	dst_d	lng	lng_rate	pos	GazeDispersionX	GazeDispersionY
0	295564.263158	1833.736842	1142.496646	78.930669	2540.699894	0.009020	0.315789	121.564592	79.115769
1	169841.466667	2865.066667	1101.699266	72.569751	3562.256635	0.021860	0.400000	105.939026	58.224502
2	53749.944444	1550.277778	1103.832872	-48.624766	1761.948955	0.056655	0.555556	95.078609	59.457161

Three scatter plots (Figure 6-12) visualize the time dimension on the x-axis mapped against three input features. No distinction between clusters is identified based on the derived measures.

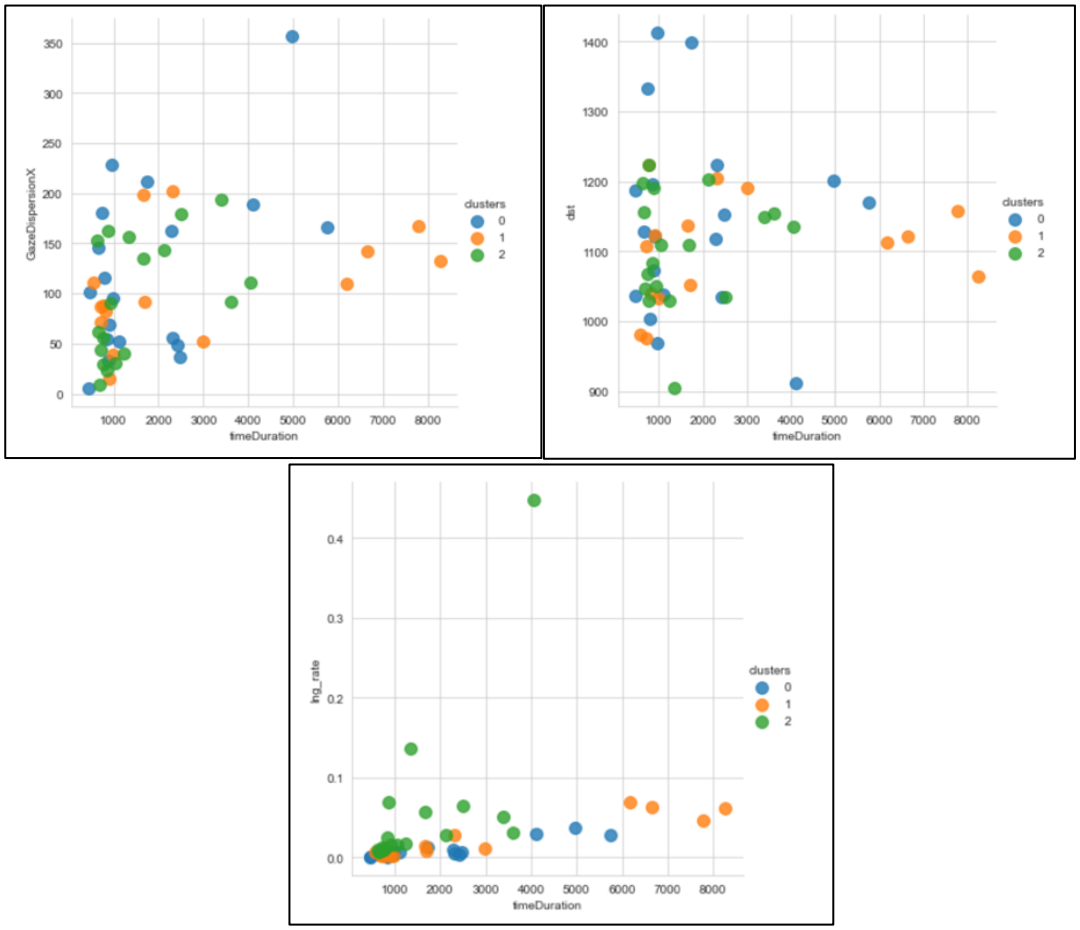


Figure 6-12. K-means clusters in gaze metrics for lights events

K-Means Model 3

A third k-means model was trained on the full 48-subject 2D visual attention feature dataset.

Table 28. K-Means 2-Cluster Parameters

clusters	timeStamp	screenGazePosX	screenGazePosY	d_travel	speed	light	Behavior	LightStatus
0	266172.340933	1096.473066	1080.574441	60.875822	1.237978	9.475105	0.757600	0.285901
1	86902.464169	1119.861665	1079.883157	54.656339	1.303882	3.431850	0.710539	0.281967

A visual interpretation of the resulting cluster scatterplots in Figure 6-13 leads to the conclusion that the clusters are not meaningfully distinct.

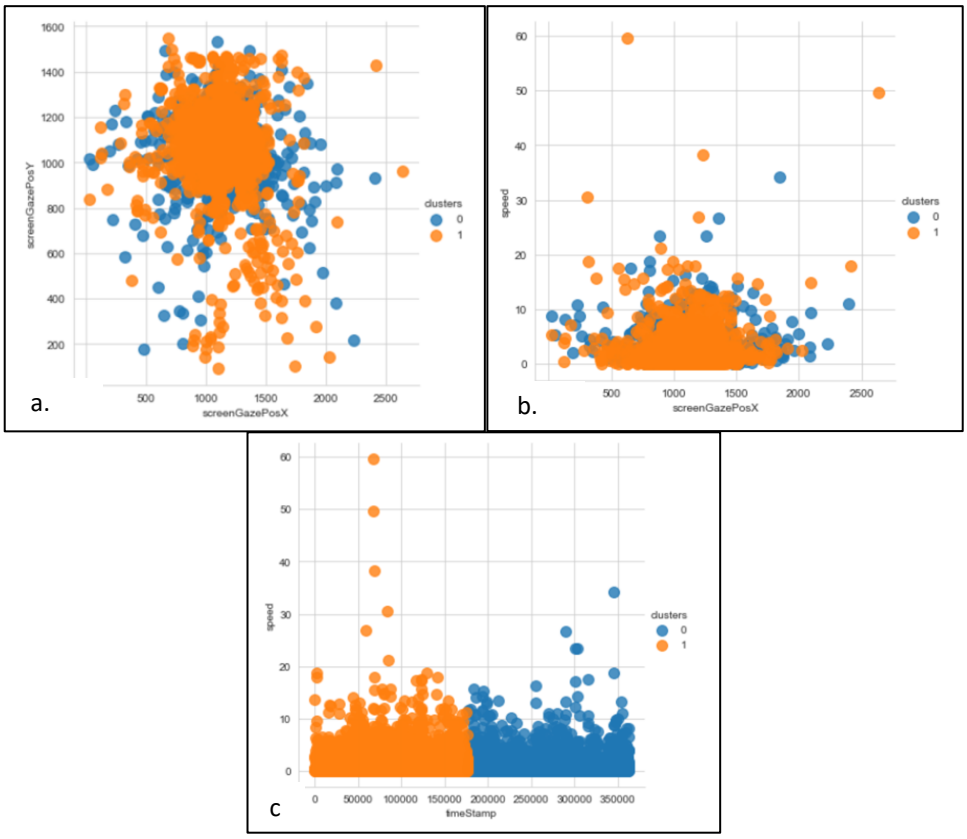


Figure 6-13. K-means clusters - Gaze Position and Speed

The clusters do stack in the speed-time dimension (c), but the overlap of clusters and even data distribution within clusters in the screen gaze position X and Y dimensions indicate there is no readily apparent cluster separation.

6.4.2 Principal Component Analysis

A principal component analysis was conducted with all features in the 2D 48-subject all-trial dataset. The resulting clusters are visualized in Figure 6-14.

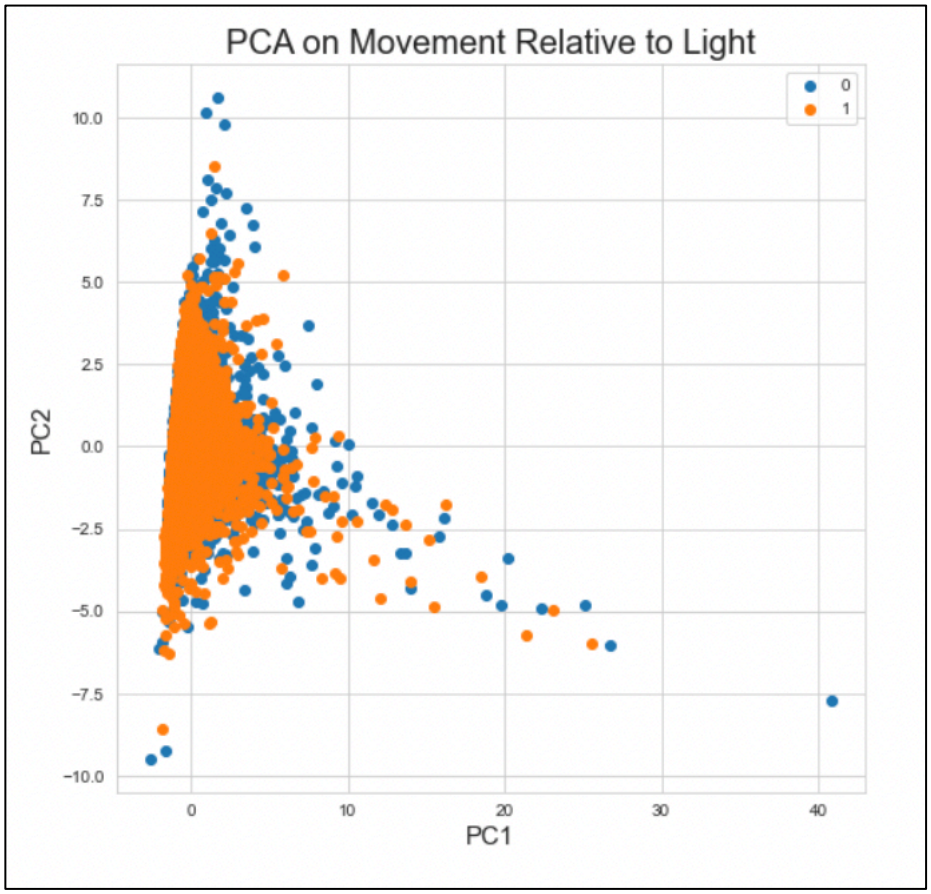


Figure 6-14. Principal Component Analysis, Attentional Shift

Principal component analysis did not yield visibly differentiable clusters when trained on all 2D features across the 1.3-million annotated gaze point dataset.

CHAPTER 07:

3D RESULTS

7 3D Results

VISUALIZATIONS AND HIGH-LEVEL OBSERVATIONS

7.1 Vector Visualizations

In figure 7-1, the direction the participant's head is rotated (red) and the direction the participant's gaze is pointing (blue) are aligned for the duration of the walk across the path (green). In the middle of the path, the head and gaze vectors both rotate across the walking trajectory, and the subject is seen to shift their gaze from looking left to looking towards the right (time increases along the x axis).

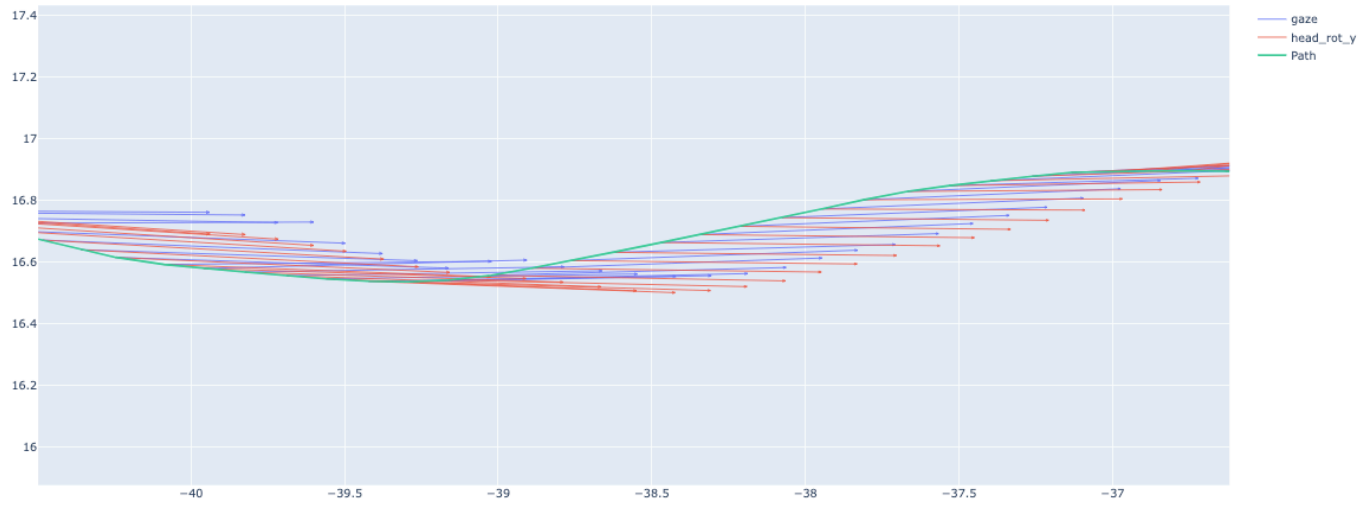


Figure 7-1. Walking view, optical angles

Figure 7-2 shows the visual attention of a participant who was standing in one location and surveying their surroundings. The orientation of the gaze and head vectors vary widely.

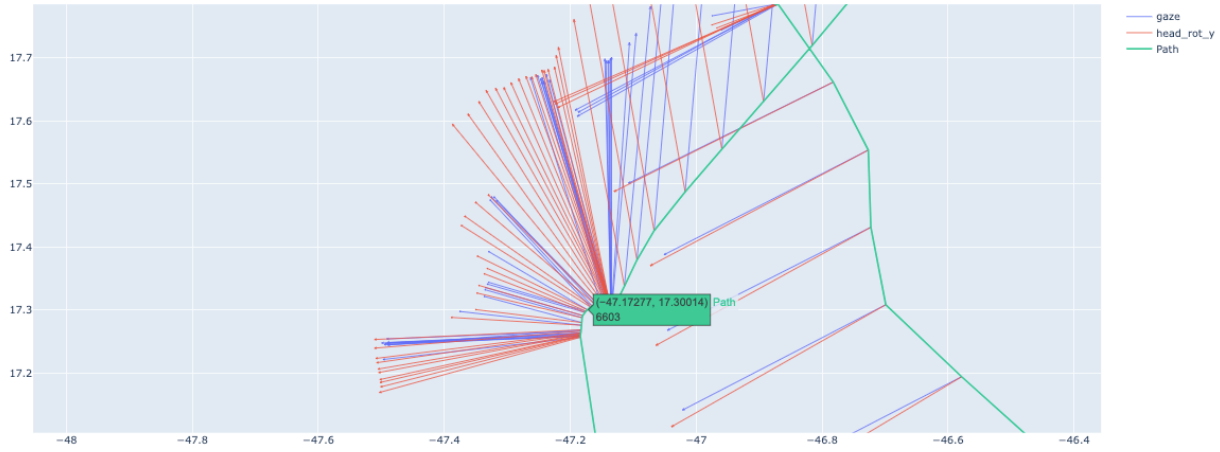


Figure 7-2. Gaze and head rotation vary widely in a corner setting

7.2 3D Visualization of all 1.34 million Gaze Points

Figure 7-3 depicts all head positions (color gradient) and gaze positions in the world view (black) for all participants, walking through all layouts, at every timestamp in the experiment. The image was generated using the VisIt visualization package (Lawrence Livermore National Labs, 2022). The color gradient affords visibility into the depth that was traversed.

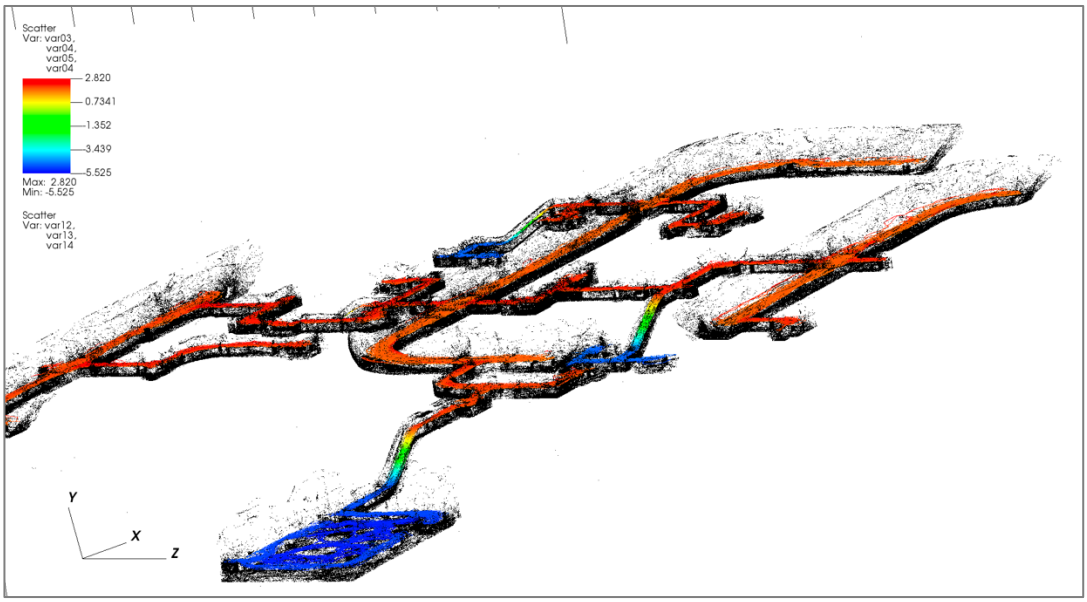


Figure 7-3. Depth in VisIt 3D Visualization All Subjects

A magnified look into the all-point VisIt visualization affords the ability to isolate individual participant paths (Figure 7-4a), and areas of concentrated visual attention allocation (Figure 7-4b). Dark bands in Figure 7-4b indicate salient areas of interest across all subjects.

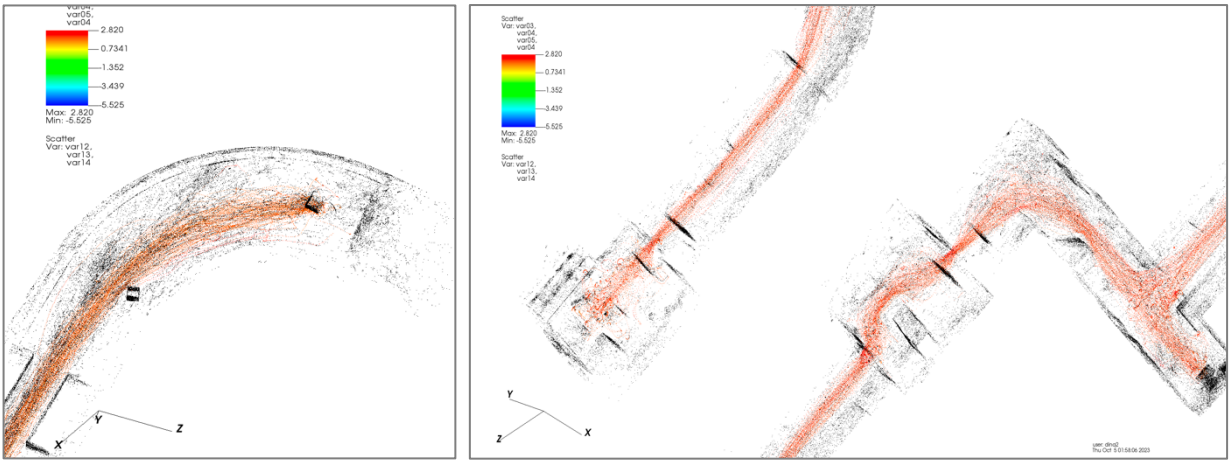


Figure 7-4. Visualization of (a) subject paths and (b) concentrated attention allocation

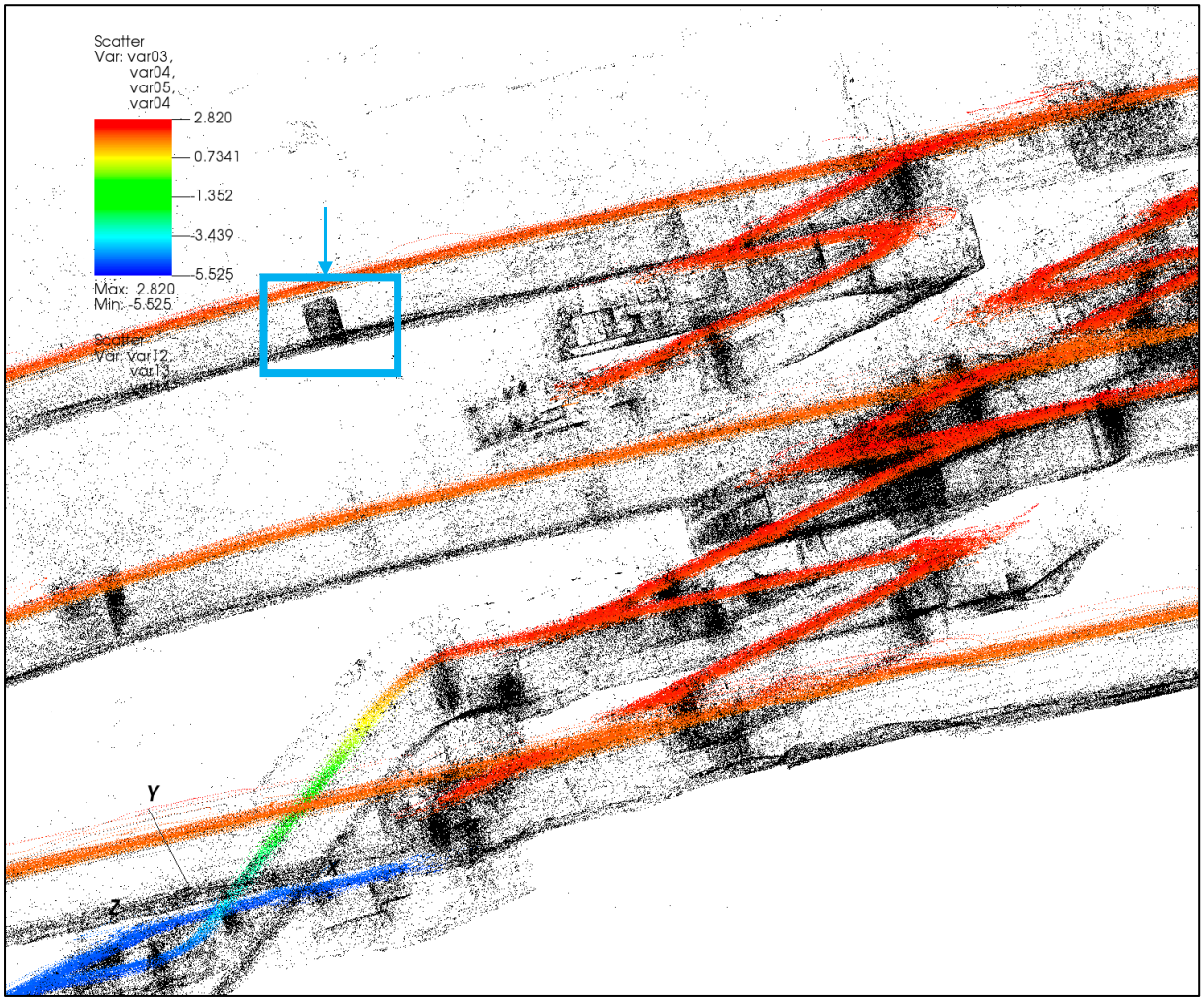


Figure 7-5. Objects in VisIt 3D Visualization

Upon closer look into the world gaze positions (black), an interesting observation emerges - the traces of objects in space upon which the gaze of multiple participants fell stand out. If an object was salient to multiple participants, as in the blue square indicated by an arrow on Figure 7-5, the collective gaze locations of multiple participants create a trace and therefore an outline of objects in space that otherwise have no other measurable indicator. The blue square in Figure 7-5 is likely the outline of one of the boxes for which the participants were searching as their primary task.

DECISION-MAKING STUDY

7.3 3D Visual Attention Patterns and Decision Events

The following sections detail the results of paired T-test comparisons of means for all native and derived features of visual attention in 3D space as they pertain to the decision-point analysis. The process of calculating and isolating 3D features from the eye-tracking data are detailed in Chapter 6, and the associated experimental design diagrams are detailed in Chapter 3. The specific hypotheses being tested are repeated before each analysis for interpretability by the reader. The following hypotheses are presented for analysis of one of the features in 3D space, the Gaze Path Length. The hypothesis space follows this same structure for each additional feature tested in this chapter.

H(o): The null hypothesis maintains that the Gaze Path Length travelled by the eye **before the decision** point is **equal** to the mean Path Length travelled **after the decision point**.

H(a_1): The first research hypothesis asserts that the mean path length travelled by the eye before the decision point is **different from** the mean path length travelled after the decision point.

H(a_2): The second research hypothesis posits that the mean path length travelled by the eye **before the decision point** is **greater than** the mean path length travelled **after the decision point.**

All subsequent analyses aim to identify 3D visual attention features which contribute significantly to discernment of steps of the decision-making process. To briefly summarize the experimental psychophysical event being analyzed: the user perceives a need for additional navigational support, the NavTools button is depressed via direct action by the user, and the Navigational Support Tools are displayed on the Heads-Up Display in the headset during the run (Zaman, 2022). The NavTools-activation is used as an indicator of a user decision-making event. This event provides a measurable signal for psychometric analysis.

Four (4) time-windows are isolated before and after the decision point and examined in the following sections for their contribution to the decision-making cognitive process.

7.4 t-Test: Paired Two Sample for Means

First, **30-second** windows are isolated as individual events **before** and **after** the participants triggered NavTools information display on the HUD (Zaman, 2022). Paired T-test comparison of means across the two windows (before and after) are conducted across all 7 3D space features for all subjects and all layouts. The table below displays a cross-section of the results of the T-test comparison the Total Path Length travelled by the gaze before the decision point for all NavTools-request events for each participant.

Table 29. T-test comparison of Derived Measures, 30sec Before/After Decision

P Values for One-Tailed T test (Greater)							
	Length of Gaze Path, Total	Angle: Head to Gaze	Speed, Angular: Head to Gaze	Change in Rotation	Head Rotation: Euler Y	Pupil Diameter	Eye Openness
1							
	0.987	0.943	0.083	0.995	0.083	0.995	0.995
	0.046	0.761	0.990	0.994	0.990	0.994	0.994
2	0.000	0.193	0.002	0.005	0.002	0.005	0.005
	0.834	0.186	0.719	0.948	0.719	0.948	0.948
	0.948	0.014	0.917	0.900	0.917	0.900	0.900
	0.936	0.460	0.978	0.524	0.978	0.524	0.524
3							
	0.371	0.296	0.547	0.718	0.547	0.718	0.718
4	0.680	0.562	0.973	0.511	0.973	0.511	0.511
	0.623	0.208	0.865	0.807	0.865	0.807	0.807
	0.050	0.042	0.102	0.277	0.102	0.277	0.277

Each numbered group indicates the results of one participant. All grey rows represent layouts which were excluded from the analysis based on one of two conditions:

(1) the presentation of the Navigation Tools was always ON throughout the layout, or

(2) no NavTools requests were ever made by the user for any reason during the layout.

Figure 7-6 presents the results of T-test comparison between 30-sec groups before and after the navigational tools request. The results are presented as percentages. It is important when interpreting Figure 7-6 to note that the two groups presented are not *before-decision* and *after-decision* conditions, instead they show the percentage of the total trials (all subjects, all layouts) where significant differences were observed in the 30-SEC BEFORE and 30-SEC AFTER conditions for all seven 3D visual attention features. Blue bars indicate significance in the before/after 30-sec windows at the 90% confidence level, and orange bars indicate significance in the before/after 30-sec windows at the 95% confidence levels.

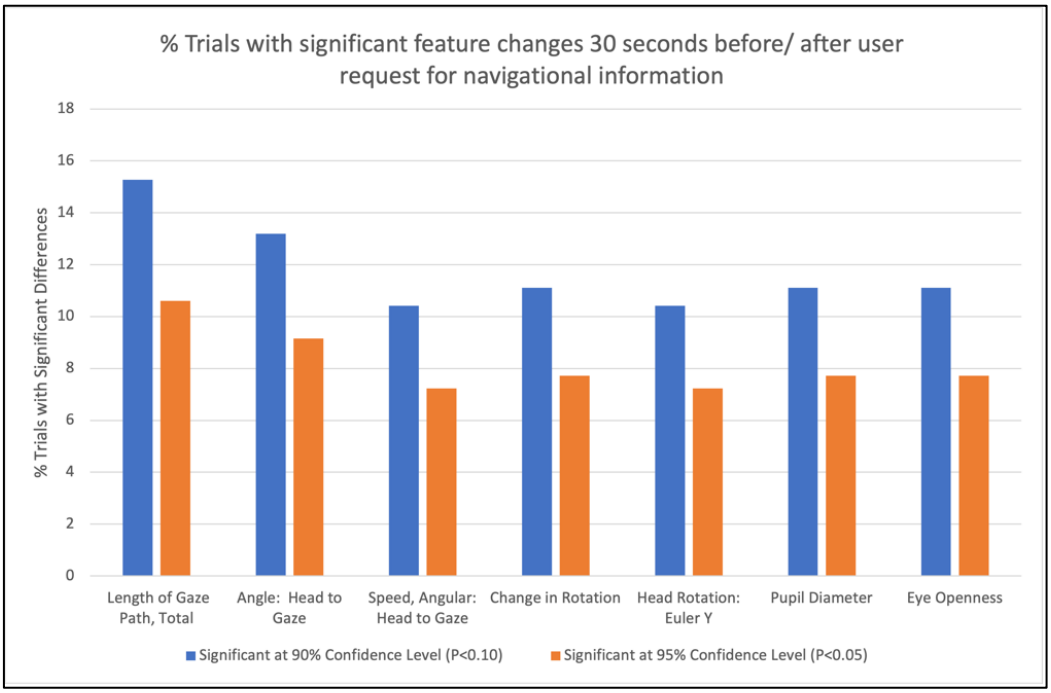


Figure 7-6. 30sec Before/After Decision Point, T-Test Comparison of 3D Features

The features that exhibited statistically significant changes across the decision point in the highest percentage of trials for the whole 48-subject group (196 layouts) were: Total Length of Gaze path Travelled in 3D space (15% of all subject runs at the 90% confidence level, and 11% at the 95 confidence level), and the Visual Angle between Head and Gaze (13% of all subject runs at the 90% confidence level, and 9% at the 95 confidence level).

7.5 ML Predicting Decision, Event-level Set

Machine learning prediction accuracies for classification of the 30-second before/after decision event windows are presented in Table 30. Models trained on the events data did not perform better than chance.

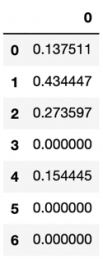
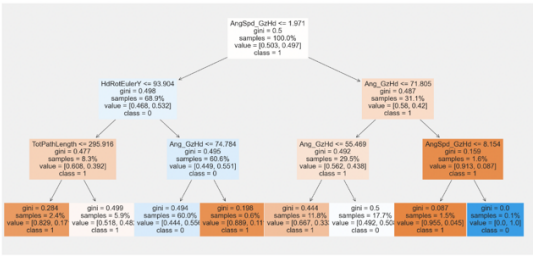
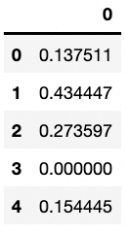
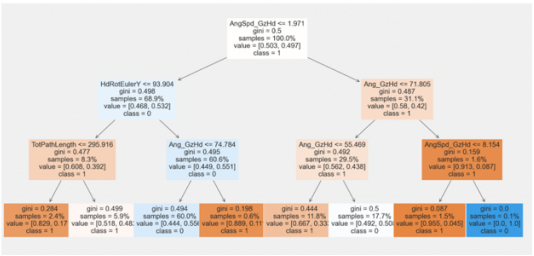
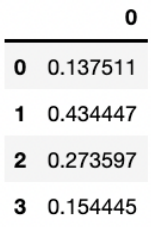
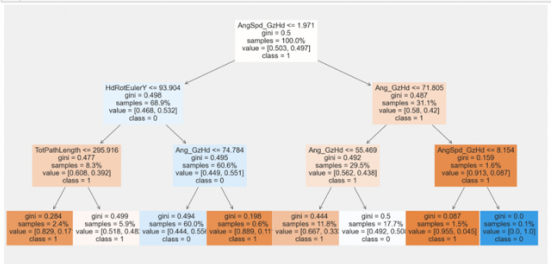
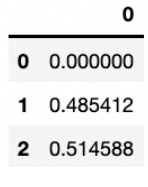
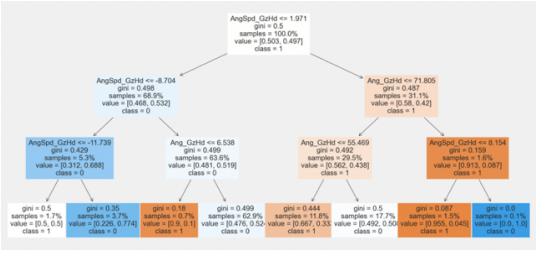
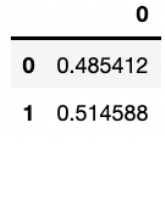
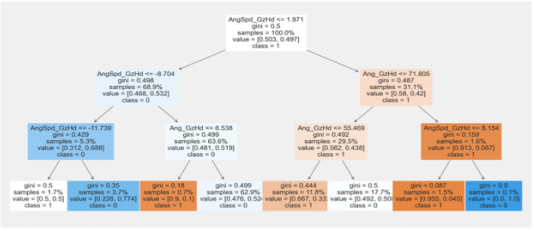
Table 30. ML Predicting Decision Event Point for 30 sec Windows

Target	Classes	Input Features	Model	Accuracy
Decision Event	1. 30 sec. before 2. 30 sec. after	Average Eye Openness	Logistic Regression	0.523
		Rotation Change	Logistic Regression	0.5
		Angular Velocity, Gaze to Head	Logistic Regression	0.521
		Angle between Head and Gaze	Logistic Regression	0.526
		Total Distance Travelled by Gaze	Logistic Regression	0.479
		1. Total Path Length 2. Average Angle Gaze to Head 3. Average Angle Speed Gaze to Head 4. Change in Y Rotation Across Event 5. Total Head Rotation Y 6. Average Pupil Diameter 7. Average Pupil Diameter	Multiple Logistic Regression	0.518
			Multiple Linear Regression	0.001
			Decision Tree	0.532
	Dataset: All 3D Nav Events Extracted from all 48 Subjects			

7.6 Candidate Elimination Process by Feature Importance

Table 31 shows the decision tree splits for the candidate elimination method for all 3D events.

Table 31. Candidate Elimination Method for Decision Tree Optimization

Model	Feature Importance	Decision Tree Splits
1		
2		
3		
4		
5		

The best predictions using the candidate elimination method (Table 32) performed only slightly better than chance, at approximately 50-51% accuracies for the decision tree, 50-52% accuracy for the KNN, and 50-52% accuracy for the logistic regression. Eliminating candidates did improve the performance of the models, however the improvements were small.

Table 32. ML Results: Candidate Elimination, 3D Decision Evens

Target: Decision Event (Before/After)								
	Model Input Features	**Multiple Logistic Regression**	**Multiple Linear Regression**	**KNN**	**Decision Tree**			
					Accuracy	**Precision**	**Recall**	**F1**
1	1. TotPathLength 2. Ang_GzHd 3. AngSpd_GzHd 4. RotChange 5. HdRotEulerY 6. avPupDiam	0.501	-0.001	0.50	0.508	0.508	**0.825**	0.629
2	1. TotPathLength 2. Ang_GzHd 3. AngSpd_GzHd 4. RotChange 5. HdRotEulerY	0.506	-0.001	0.506	0.506	0.508	**0.825**	0.629
3	1. TotPathLength 2. Ang_GzHd 3. AngSpd_GzHd 4. HdRotEulerY	0.511	-0.001	0.511	0.508	0.508	**0.825**	0.629
4	1. TotPathLength 2. Ang_GzHd 3. AngSpd_GzHd	0.516	-0.001	0.516	0.505	0.506	**0.876**	0.642
5	1. Ang_GzHd 2. AngSpd_GzHd	0.521	0.000	0.521	0.505	0.506	**0.876**	0.642

7.7 T-Test Comparison of Means for 3D Visual Attention Markers

7.7.1 7-Second Before/After Decision Point

A t-test comparison of means revealed significant differences in three (3) of the derived 3D attention features across the two 7-second windows before and after the NavTools-request decision point (Table 33).

Table 33. T-Test 3D Derived Features: 7-sec Before/After Decision Point

3D Full Individual Gaze Point Database (derived measures): 7 SECOND BEFORE/ AFTER						
7 SEC BEFORE (Gaze Point Count: 99,168). 7 SEC AFTER (Gaze Point Count: 75,390).						
	Means		**Standard Deviations**		**T-Test Result**	
Feature	**Before Mean**	**After Mean**	**Before StdDv**	**After StdDv**	**T Statistic**	**P Value**
Change in head direction	-0.03	-0.01	3.04	2.62	-1.56	0.119
Speed of Change in head direction	-0.61	-0.26	42.11	36.68	-1.80	**0.072**
Gaze Path Length	1.09	1.08	5.78	4.88	-0.33	0.744
Angle Gaze to Head	0.10	-2.02	37.35	37.61	11.76	< 0.001
Gaze Speed	20.08	19.45	118.01	94.53	1.20	0.23
Step Length	0.105	0.097	0.20	0.22	7.48	< 0.001
Walk Speed	1.77	1.60	1.22	1.27	28.44	< 0.001

The **Angle between Gaze Direction and Head Direction** is significantly different between the Before condition (M = 0.10, σ = 37.35) and After condition (M = -2.02, σ = 37.61) with a T-test statistic of 11.76 and p-value < 0.001. Although the change in means indicates a decrease in angle, the magnitude of the angle in the After condition is higher than in the window preceding the decision point. Therefore, the difference can best be interpreted as an increase in angle magnitude and a change in direction (head turns before gaze vs. gaze turns before head).

Step Length decreased significantly between the Before condition (M = 0.105, σ =0.097) and the After condition (M = 0.097, σ =0.22), with a T-test statistic of 7.48 and associated p-value < 0.001.

Walk Speed decreased significantly between the Before condition (M = 1.77, σ =1.60) and the

After condition (M = 1.22, σ =1.27), with a T-test statistic of 28.44 and associated p-value < 0.001. The T-test comparison of means found ***no significant difference*** in ***Gaze Speed, Gaze Path Length, Change in Head Direction, or Speed of change in head direction*** between the 7-second windows before-decision and after-decision.

7.7.2 3-Seconds, 7-Second, and 15-Seconds Before/After Decision Point

Next, paired T-tests were performed to evaluate the differences in all derived 3D visual metrics across multiple time windows around the decision point. Once again, all decision-point events were extracted from the aggregate dataset of all trials for 48 subjects. The change in all 7 features across 3-sec, 7-sec, and 15-sec timeframes before and after the NavTools request decision were evaluated.

Table 34. T-test Result: 3D Derived Features, 3s, 7s, 15s, Before/After Decision

All 3D Individual Gaze Points Database, All Time brackets, (derived measures) T-Test Result						
BEFORE vs. AFTER DECISION EVENT						
	3 SEC		**7 SEC**		**15 SEC**	
Feature	**T Statistic**	**P Value**	**T Statistic**	**P Value**	**T Statistic**	**P Value**
Change in head direction	0.72	0.471	-1.56	0.119	-1.65	0.100
Speed of Change in head direction	3.04	0.002	-1.80	**0.072**	-2.90	0.004
Gaze Path Length	2.07	0.038	-0.33	0.744	0.97	0.332
Angle Gaze to Head	6.57	< 0.001	11.76	< 0.001	10.28	< 0.001
Gaze Speed	2.60	0.009	1.20	0.23	1.85	**0.064**
Step Length	7.69	< 0.001	7.48	< 0.001	0.43	0.663
Walk Speed	28.65	< 0.001	28.44	< 0.001	8.98	< 0.001

Three features exhibited statistically significant differences across all time windows tested: the walk speed, angle between gaze direction and head orientation, and the gaze path length calculated in 3 dimensions. These features can be consistently isolated as markers of decision-making across any time window. The time-window which saw significant changes across the highest number of features was the 3-second window.

All native features were compared across three (3) before-decision and after-decision windows: 3-sec, 7-sec, and 15-sec windows. Paired T-Test comparisons of means yielded significant differences for all features highlighted in green in Table 35.

Table 35. T-test Result: 3D Native Features, 3s, 7s, 15s, Before/After Decision

T-TEST COMPARISON OF MEANS, BEFORE/ AFTER DECISION **Significant Differences at Alpha = 0.05**			
3D Full Individual Gaze Point Database (Native Features)			
Feature	**3 SEC**	**7 SEC**	**15 SEC**
Head Rotation X (degrees)	< 0.05	< 0.05	> 0.05
Head Rotation Y (degrees)	< 0.05	< 0.05	< 0.05
Head Rotation Z (degrees)	< 0.05	< 0.05	< 0.05
Eye Openness (average)	< 0.05	< 0.05	< 0.05
Pupil Diameter	< 0.05	< 0.05	< 0.05
World Gaze Position X (m)	> 0.05	< 0.05	< 0.05
World Gaze Position Y (m)	< 0.05	< 0.05	< 0.05
World Gaze Position Z (m)	< 0.05	< 0.05	< 0.05
Head Position X (m)	> 0.05	< 0.05	< 0.05
Head Position Y (m)	> 0.05	< 0.05	< 0.05
Head Position Z (m)	< 0.05	< 0.05	< 0.05
Screen Gaze Position X (px)	< 0.05	< 0.05	< 0.05
Screen Gaze Position Y (px)	< 0.05	< 0.05	< 0.05
Screen Gaze Position Z (m)	< 0.05	< 0.05	< 0.05

Significant differences were identified between Before-Decision Point and After-Decision Point groups in three time-windows (3-sec, 7-sec, and 15-sec before and after the decision event) at the $p<0.05$ level. Features with statistically significant differences are highlighted in green, whereas features which saw no significant differences across before and after windows are highlighted in red. For the 7-second window, every feature saw a statistically significant difference across the decision point. The 3-second window saw the most features with no significant change across the decision point of the time windows assessed.

One additional behavior was isolated, the speed of walking, and T-test analyses were performed to identify changes in features of visual attention across speed classes. The full 1.34-million observation aggregated dataset was classified into two speeds (fast and slow), where "fast" walking speeds were considered above the mean and "slow" walking speeds were considered below the mean for each group. Features with statistically significant differences are highlighted in green in Table 36, whereas features which saw no significant differences across before and after windows are highlighted in red.

Table 36. T-test Result: 3D Native Features, 3s, 7s, 15s, Slow/Fast Walking Comparison

T-TEST Comparison Of Means, SLOW / FAST Conditions Significant Differences at Alpha = 0.05			
3D Isolated Events Dataset, 48-Subject			
Feature	**3 SEC (m=1.48 m/s)**	**7 SEC (m=1.59 m/s)**	**15 SEC (m=1.72m/s)**
Gaze Dispersion X	< 0.05	< 0.05	< 0.05
Gaze Dispersion Z	< 0.05	< 0.05	< 0.05
Angle Between Gaze/ Head	< 0.05	< 0.05	< 0.05
Angle Speed, Gaze to Head	> 0.05	< 0.05	> 0.05
Total Path Length	> 0.05	< 0.05	< 0.05
Change in Head Rotation	< 0.05	< 0.05	< 0.05
Head Rotation Euler Y	> 0.05	> 0.05	> 0.05
Avg. Pupil Diameter	> 0.05	> 0.05	> 0.05
Avg. Eye Openness	< 0.05	< 0.05	< 0.05

Significant differences were identified between the **slow-walk** and **fast-walk** groups across three time-windows (3-sec, 7-sec, and 15-sec before and after the decision event) at the $p<0.05$ level. Gaze Dispersion in the X and Z direction, Angles between gaze and head orientation, the change in Head Rotation, the average Eye Openness exhibited statistically significant differences between the slow-walking and fast-walking conditions in all time windows. In the 7-second window, the angle speed and total path length also changed significantly between speed classes.

7.8 ML Predicting Decision, Full Dataset

The model trained on features down-selected via the T-test for significance performed with **70% accuracy**. The Breakdown of classification nodes and leaves is seen in Figure 7-7 .

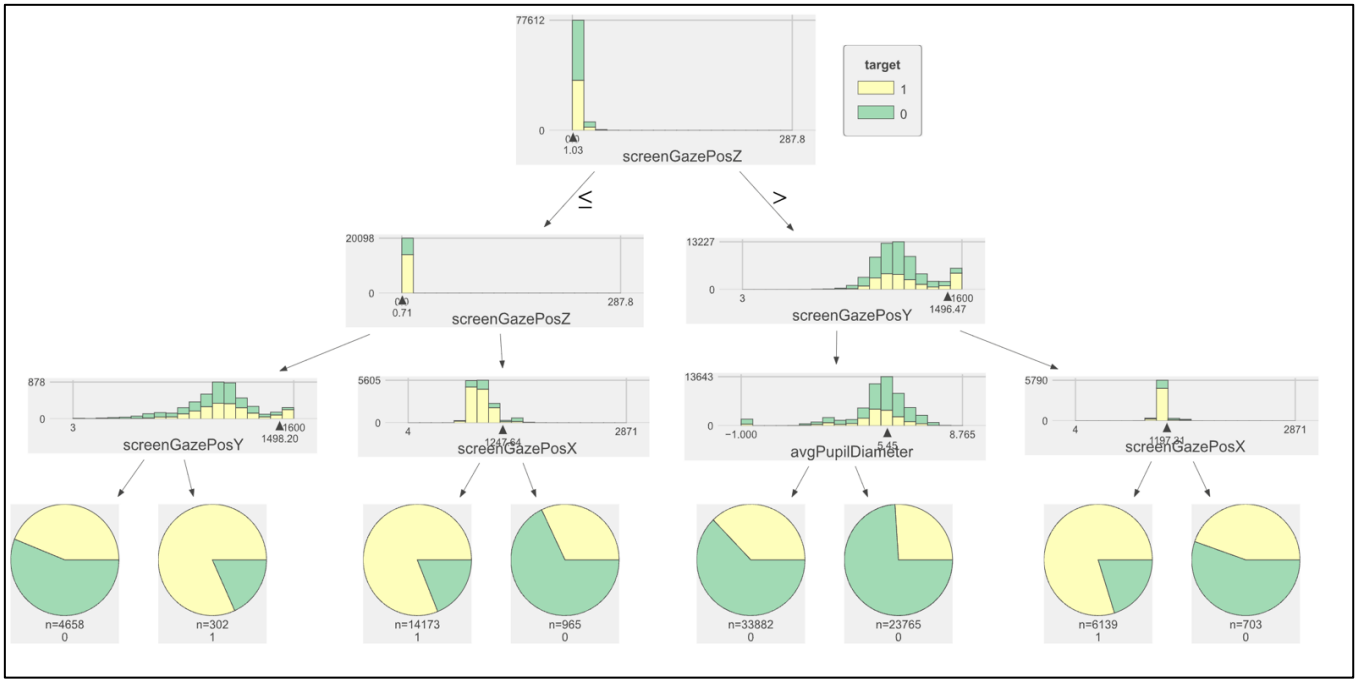


Figure 7-7. 3D 7sec Before/After Classification, Decision Tree after Feature Selection

Note: Algorithm Copyright (c) 2021 Terence Parr. https://github.com/parrt/dtreeviz

The 70% accuracy of the decision-tree trained exclusively on features that changed significantly across classes (as identified using the T-test method) surpassed the accuracies seen for models trained and tuned on features isolated via the candidate elimination method. The best predictions using features down-selected using the candidate elimination method performed only slightly better than chance, at approximately 50-51% accuracies.

7.9 Unsupervised Learning: K-Means Clustering on 3D Dataset

Tables 37 - 39 show the results to K-Means analyses performed on the 3D feature space. Each table value indicates the mean of the cluster identified. Each table of cluster means is followed by a visualization of the respective clusters.

Table 37. K-Means 3-Cluster Analysis on 3D Feature Space

clusters	TotPathLength	Ang_GzHd	AngSpd_GzHd	RotChange	HdRotEulerY	avPupDiam	avgOpen
0	348.791334	55.724033	-0.581008	3.011925	182.649147	5.016860	0.914098
1	815.825917	54.619286	-0.024189	2.324118	187.175437	5.133357	0.918607
2	1808.006802	54.294244	-0.395467	1.898388	138.241810	5.390155	0.941960

Data is clustered into three (3) clusters. The clusters are visualized as distinctly separated in the Total Path Length dimensions in Figure 7-8.

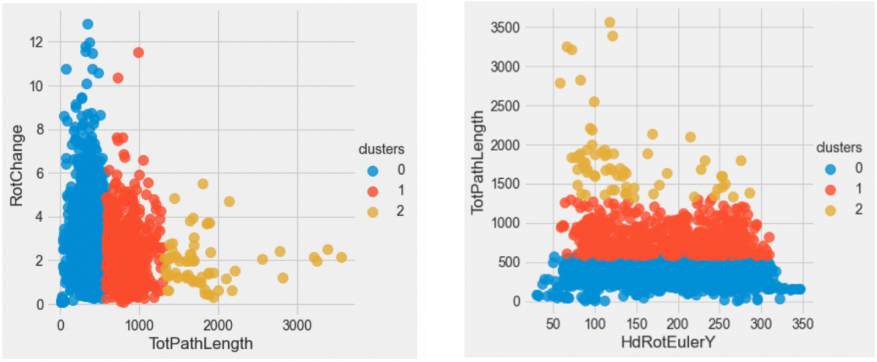


Figure 7-8. 3D K-Means Clusters Visualization

Table 38. K-Means 6-Cluster Analysis on 3D Feature Space

clusters	TotPathLength	Ang_GzHd	AngSpd_GzHd	RotChange	HdRotEulerY	avPupDiam	avgOpen
0	251.435836	54.598136	-0.683211	3.062355	183.737707	4.990293	0.915682
1	1676.589998	54.034938	-0.561746	1.889279	140.569609	5.376379	0.942387
2	1001.003740	51.930374	-0.195198	2.232929	176.403330	5.130570	0.914141
3	455.060452	56.970920	-0.474784	2.957679	181.382010	5.044283	0.912222
4	3084.194264	56.814693	0.704090	2.055215	88.163797	5.404741	0.936482
5	704.057901	56.261804	0.092115	2.377068	194.162396	5.141896	0.921847

The 6-cluster analysis resulted in distinct clusters along the Total Path Length dimension, as visualized in figure 7-9a. Clustering in the Head Rotation and Angle Speed dimensions did not result in distinct cluster means, as seen in Figure 7-9b.

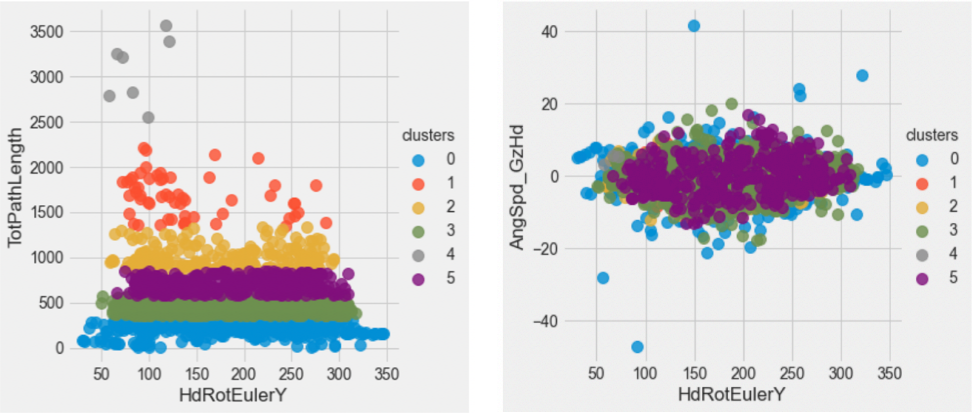


Figure 7-9. 3D K-Means 6-Cluster Visualization of head rotation vs. (a) total path, and (b) angle speed

Table 39 and associated visualizations in Figure 7-10 show the numeric and visual results of unsupervised clustering with 3-clusters trained on seven features in 3D space.

Table 39. K-Means 3-Cluster Analysis for 3D Derived Features

clusters	headRotEulerY	avgPupilDiameter	avgOpenness	ChangeHeadRotation	Speed_HeadRotEulerY	AngleGazetoHead	AngleSpeed_GzHd
0	183.639914	5.204590	0.874981	-0.114893	-2.992277	80.284744	-362.496432
1	261.054174	5.300105	0.915817	0.040926	0.649146	97.473214	19.143185
2	74.262491	5.265253	0.914506	-0.031954	-0.472746	64.559616	13.870824

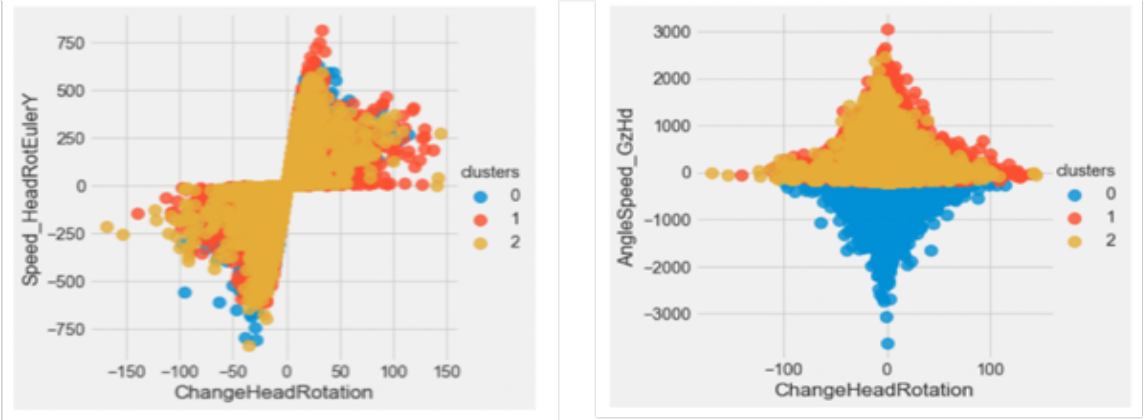


Figure 7-10. 3D K-Means Clusters Visualization

CHAPTER 08:

DISCUSSION

8 Discussion

In this Chapter, nuances and significance of the methods developed and the implications of study results are discussed within the context of initial research questions and broader literature scope. Methods of extracting markers of visual attention were developed in both 2-dimensional and 3-dimensional projections of the Virtual Reality space. Statistical methods and machine learning were applied to study cognitive and behavioral events from attention patterns.

8.1 Study Methodology in 2D and 3D Geometry

Through the process of establishing algorithmic functions for classification of visual attention in 3D space, several challenges were encountered and overcome. Unlike analyses in the 2D screen-plane, which relied on a pre-defined coordinate system, working with gaze in motion in the 3D space required establishing dynamic points of reference to afford the study of orientation. As the user navigated through the VR space, the point of reference for each visual angle calculation was not a fixed point in the 3-Dimentional space. Instead, the origin point, set as the position of the head in VR space, was algorithmically defined to adapt continuously with the participant's movement. The participant's line of sight was mapped and continuously updated by extending a vector from each visual reference point (head position) towards the intersection points of the line of sight with objects in the spaces traversed.

The extension of vectors from the position of the head into the Z-coordinates of the screen plane afforded the possibility of isolating the orientation of the head at every point while the participant was walking in the trial. Finally, a simple distance formula filter applied continuously between any two points of the participant's iterating head location yielded the path of motion, or the walking direction. The associated angles between these three vectors could then be derived by simple trigonometric functions. All line-of-sight vectors, head rotation vectors, direction of motion, and derived angles across all movements through 3D space were extracted and analyzed as markers of human visual attention.

8.2 Study Findings

In this Thesis work, supervised studies were conducted in (4) Research Areas:

1. Relationships between Response Latency and 2D Perception Markers
2. Attentional Shift Behaviors towards Stimulus
3. Differentiating Between Presence and Absence of Stimulus
4. Predicting Decision-Making

Unbiased exploratory open inquiry was also conducted using unsupervised ML modeling.

Testable null and alternative hypotheses were defined for each category of study and tested using statistical techniques. The first three Research Areas (concerning response time, attention shift, and stimulus) were explored in the 2D Space using the on-screen secondary task Light as a stimulus for the study of attention. The fourth question, decision-making, was studied in 3D Space using the decision to request navigational support as the psychological event of interest.

Study 1: 2D Gaze Geometry Study

The first study sought to identify correlations between baseline visual parameters in the 2D plane and cognitive load, measured as the latency in the light task response time. The distance between the on-screen gaze center and the light stimulus was visualized for every isolated light event. The preliminary one-subject analysis and the 20-subject linear regression found no correlation between the gaze-to-stimulus distance and response time latency. No signal was identified, therefore the null hypothesis cannot be rejected. The absence of a correlation indicated that proximity of line of sight to the stimulus alone does not explain the mechanism by which the participant notices and extinguishes the light. This prompted further investigation into the remaining 2D visual attention features to understand their contribution to the mechanism of visual perception. A small exploratory preliminary study was also performed to classify ocular movements by speed. A KNN model was trained to classify Fixations vs. Saccades with 90% accuracy. More sophisticated static and dynamic AI-assisted classifications of fixation and saccadic movements have been developed and should be considered in future studies (Enders,2021).

Study 2: Attentional Shift Study

The second study conducted in the context of 2D attention sought to understand the motion of attention during the process of extinguishing the light. If the task response was accompanied by movement of the gaze towards the light, it can be inferred that subjects actively relied on overt attention to perceive the stimulus. If, however, the visual attention window does not shift towards the light stimulus over the course of the light-task event, the signal is likely to be perceived peripherally.

A decision tree first classified the two states (moved towards / moved away from light) with 95% accuracy when the *change in distance* metric was included in the input features. However, as the attentional shift metric was derived directly from the change in distance, this finding was not in itself meaningful. Once *change in distance* was removed from the feature set, however, decision tree modeling afforded analysis of the contribution of remining features to the attentional shift. The greatest contribution to attentional shift identified in the preliminary study is made by the path gaze length, followed by the gaze speed across the event.

Logistic modeling did not accurately predict movement behavior relative to stimulus from total gaze speed and path travelled across light events. The null hypothesis was not rejected for Research Question 3: modeling results did not identify movement towards or away from the stimulus over the course of the light-response event. This suggests that perception of the light stimulus relied on mechanisms of covert attention instead of overt attention, since the eye did not directly saccade towards the stimulus to bring the signal into the window of visual attention.

Study 3: Presence and Absence of Stimulus: (Light ON/ Light OFF study)

A third study in the 2-dimensional space was conducted using the million-plus gaze dataset aggregated from the original experiment with 48 participants and 4 layouts (Zaman et. al, 2020) and new derived features, annotated with light status condition. The aim of the study was to identify features of 2D attention which would indicate a behavioral change correlated with the presence of the stimulus (and therefore the presence of the secondary task and increased cognitive load). Six features exhibited statistically significant changes between the two conditions (Light ON and Light OFF). Pupil diameter and eye openness increased significantly at the onset

of the stimulus. Head rotation *decreased* significantly during the secondary task in the X and Y directions but *increased* in the Z direction. The distance travelled by the gaze in the 2D screen plane was significantly lower during the light-ON events, indicating that the gaze point was closer to the stimulus light during the times when the light was successfully extinguished than in the absence of stimulus. The significant differences observed between features of visual attention associated with the two light status groups *support the alternative hypothesis* proposed for Research Question 4: Differentiating between Light ON vs Light OFF events. As these features differ significantly between cognitive light On/Off states, the results build confidence that a machine learning model can be trained to predict the presence of stimulus. KNN modeling results also supported the alternative hypothesis proposed for Research Question 4. The model distinguished between the presence and absence of the stimulus light with a 70% accuracy rate. Further study may consider different accuracy levels and different validation metrics for model validation.

Study 4: Decision Study in 3D Space

For the decision study, time windows of four different lengths were isolated before and after the user-triggered Navigational Tool was displayed on the HUD. The time at which the NavTools were displayed was isolated as the user response to their perceived need for navigational support. In other words, the user decision to request NavTools was an indicator of the user knowing that they were lost and in need of additional guidance. The timestamp of NavTools activation is considered the "decision time" for this analysis, and the features of visual attention at several time windows before and after this decision-making event are analyzed.

Paired T-tests were used to evaluate the differences across the Decision Point (before/after classes) for all 3D visual metrics. All decision-point events were extracted from the aggregate dataset of all relevant 196 trials across 48 subjects via two automated scripts. The first analysis identified statistically significant changes in all 3D features between the 30-second before-decision window and 30-second after-decision window. The features of visual attention which changed in the highest percentage of trials for the whole 48-subject group (196 layouts)

over the 30-second windows were: Total Length of Gaze path Travelled in 3D space and the Visual Angle between Head and Gaze.

In the second T-test analysis, all native and derived features were evaluated for significantly different changes across 3-sec, 7-sec, and 15-sec timeframes before and after the NavTools request decision. The 7-second window performed the best across native features in the 3-sec, 15-sec, 7-sec, and 30-sec windows explored as it yielded the most native features with significant differences across the two conditions: before decision and after decision. Three derived features exhibited statistically significant differences across the 3, 7, and 15-second windows: walk speed, angle between gaze direction and head orientation, and the gaze path length calculated in 3 dimensions. The implication of these findings is that these three features can be consistently isolated as markers of decision-making.

Multiple supervised and unsupervised machine learning models were constructed and trained on a combination of native and derived visual attention features in the 3D space to predict the moment of user need for navigation support. The best-performing model predicted the decision-point with 70% accuracy. The decision-tree was trained exclusively on features that first yielded significant differences between before/after conditions (using the T-test method). Models trained by this method surpassed the accuracies seen for models trained and tuned on features isolated via the candidate elimination method. The best predictions using features down selected using the candidate elimination method performed only slightly better than chance, at approximately 50-51% accuracies.

Based on this result, the null hypothesis of Research Question 5 is rejected. The decision tree classification accuracy of 70% *supports the alternative hypothesis*: a machine learning model can in fact be constructed and trained on visual attention features to accurately predict the moment when a user needs navigation support at a rate better than random chance. Future studies might consider setting a higher accuracy threshold for validating the success of the model.

8.3 Methodological Contributions

Algorithmic methods were developed to automatically process comparisons of features, from all participants and experimental runs. The first method involved some manual steps and was more descriptive at the individual level, while the second method was significantly faster to execute across all features but lacked nuance at the level of individual participants. Overall, preliminary T-test selection of significant features proved to be a data pre-processing method which generated the highest accuracy ML model predictive performance. The candidate elimination method was explored but yielded less accurate results.

One important methodological finding came in the analysis of events isolated from the raw datasets. Most ML analyses relied on isolating events for classification purposes. At first, descriptive statistics were derived from the events data and used as features to train predictions. While event-level statistics were accurate and useful in extracting insights, models trained with this data did not perform better than chance. The cause of poor model performance was identified as a loss in data resolution. The level of granularity was reduced for each event, and variability in the raw data was no longer discernible to the ML models. To resolve this, the author expanded the algorithms to isolate and extract events, encode the target variables with meaningful classifications as appropriate for each study, and then propagate the annotated classes back onto the raw dataset, resulting in fully robust training data with native features, derived features, and meaningful, tailored annotations.

CHAPTER 09:
CONCLUSIONS

9 Conclusions and Future Work

This thesis generated scientific and methodologic contributions in the space of visual attention allocation. The author developed automated Machine Learning-enabled methods of predicting cognitive process from 2D and 3D geometrical patterns of visual attention allocation.

This work yielded machine learning models that classified cognitive states with various levels of success from patterns in 2D and 3D visual attention. Hypothesis-driven studies and their respective results aided the selection and validation of meaningful classification categories and classification boundaries. The resulting models and study findings in this thesis work are intended to inform the design of smart head-mounted displays. One application of this system could read in a gaze pattern and use predictive analytics to assess the user's cognitive state. As a result, an adaptive display could vary the type and amount of information presented to be appropriate for the cognitive state. Gaze data collected from multiple people over time may aid in identification of salient objects in 3D and identify locations of interest by creating outlines of objects in space that lack measurable indicators. The gaze pattern-recognition machine learning models and data engineering methodology developed in this thesis have applications into multiple spaces. Models were trained on large VR datasets to predict cognitive markers such as decision points, reaction times, and shifts in attention. Extending the ML feature set to include lighting effects on vision in low luminosity spaces would render this technology applicable to more varied conditions.

The 3D visualizations conducted in this work show the potential for AI-enhanced eye tracking technology to support object detection in visual search and navigation applications. The studies conducted here showed the viability of eye-tracking-enabled Virtual Reality technology to provide insights into mechanisms of human attention. However, while this work generated a method of studying the cognitive processes that accompany visual attention behaviors, more work must be done to understand the resolution of the window of attention (Intriligator & Cavanagh, 2001). More focus should also be placed on the study of internal process of discernment between multisensory sources of information. Studies might be designed to analyze the mechanism of pre-saccadic attention shifts with greater resolution, to better understand the process of anticipation that precedes visual motion (Intriligator & Cavanagh, 2001).

The use of walking and search activities in the virtual environment provided a simulated mechanism of studying attention in motion. This work demonstrates that methods of visual decomposition in the field of view such as those developed by Warren and Hannon to study optical flow can be successfully applied to the study of human attention in motion using Virtual Reality technology (Warren & Hannon, 1988). The algorithmic methods established here can be extended to augmented reality glasses such as HoloLens which already assist real-world navigation in remote locations. Other applications of the ML-enhanced VR eye tracking algorithms developed in this work could aid in therapeutic rehabilitation. Immersive VR experiences could be personalized to the needs of individuals undergoing motor skill and cognitive therapy. In a modern world saturated with information, the author hopes that this technology will serve to bring clarity and a deeper understanding of the inner workings of the mind when it is needed most.

CHAPTER 10:

APPENDIX

10 APPENDIX

10.1 Appendix 1: 2D Linear Regression Semi-Automated 20X

```
#!/usr/bin/env python
# coding: utf-8

# SUBJECT 3  - 0037YF

#Importing Libraries

import numpy as np
import pandas as pd
import numpy as np
import seaborn as sns

from matplotlib import pyplot as plt

from sklearn.linear_model import LinearRegression

import warnings
warnings.filterwarnings('ignore')

## Load data ###############################################

# 1. Load Data for All Layouts

gaze1 = pd.read_csv('0037YF; 05-13-2021_18.07; EyeHeadTracking.txt')
light1 = pd.read_csv('0037YF; 05-13-2021_18.07; LightTask.txt')

gaze2 = pd.read_csv('0037YF; 05-13-2021_18.30; EyeHeadTracking.txt')
light2 = pd.read_csv('0037YF; 05-13-2021_18.30; LightTask.txt')

gaze3 = pd.read_csv('0037YF; 05-13-2021_18.53; EyeHeadTracking.txt')
light3 = pd.read_csv('0037YF; 05-13-2021_18.53; LightTask.txt')

gaze4 = pd.read_csv('0037YF; 05-13-2021_19.12; EyeHeadTracking.txt')
light4 = pd.read_csv('0037YF; 05-13-2021_19.12; LightTask.txt')

#2.  Define light position on screen

#(from Top-Left of sctreen)

# Define light position on screen, froom Top-Left of sctreen

ox=2217 # X coordinate of light point from Left of screen
oy=1181 # Y coordinate of light point from Top of screen


# 3. Clean data

# Keep only Time stamp, duration and Gaze coordinates
gaze1 = gaze1[['timeStamp', 'intervalDuration', 'screenGazePosX',
               'screenGazePosY']]

# Drop gaze points outside of screen
gaze1.drop(gaze1[(gaze1.screenGazePosX>2880) |
                 (gaze1.screenGazePosX<0) |
                 (gaze1.screenGazePosY>1600) |
                 (gaze1.screenGazePosY<0)]
           .index, inplace=True)


# Keep only Time stamp, duration and Gaze coordinates
gaze2 = gaze2[['timeStamp', 'intervalDuration', 'screenGazePosX',
               'screenGazePosY']]

# Drop gaze points outside of screen
gaze2.drop(gaze2[(gaze2.screenGazePosX>2880) |
                 (gaze2.screenGazePosX<0) |
                 (gaze2.screenGazePosY>1600) |
                 (gaze2.screenGazePosY<0)]
           .index, inplace=True)


# Keep only Time stamp, duration and Gaze coordinates
gaze3 = gaze3[['timeStamp', 'intervalDuration', 'screenGazePosX',
               'screenGazePosY']]

gaze3.drop(gaze3[(gaze3.screenGazePosX>2880) |
                 (gaze3.screenGazePosX<0) |
                 (gaze3.screenGazePosY>1600) |
                 (gaze3.screenGazePosY<0)]
           .index, inplace=True)


# Keep only Time stamp, duration and Gaze coordinates
gaze4 = gaze4[['timeStamp', 'intervalDuration', 'screenGazePosX',
               'screenGazePosY']]

gaze4.drop(gaze4[(gaze4.screenGazePosX>2880) |
                 (gaze4.screenGazePosX<0) |
                 (gaze4.screenGazePosY>1600) |
                 (gaze4.screenGazePosY<0)]
           .index, inplace=True)

# ##  4. Calculate Average Distance to Light

# For each light event in  LightTask file
# Calculate average distance from every gaze point to light point
# Calculate length of gaze path
# Calculate only for correct light events ('lightCorrectness' == TRUE)

#LAYOUT ONE
```

```
        # to light point
        light3.dst[i]=flt3.dst.mean()

        # Save to Light Task dataframe the length of gaze path
        light3.lng[i]=np.sqrt((flt3.dx**2+flt3.dy**2)).sum()

#LAYOUT FOUR

# Add empty columns to light task dataframe, for avergae distance 'dst',
# and gaze path length 'lng'
light4[['dst', 'lng']] = np.nan

for i in range(len(light4.index)):
    if light4.iloc[i]['lightCorrectness']: # only correct light events

         # start time stamp of gaze path, at start time stamp of light event
        start4 = light4.iloc[i][0]

         # end time stamp of gaze path, at start time stamp of light event
         # + Duration of light event
        end4 = light4.iloc[i][0] + light4.iloc[i][1]

        # Make temporary dataframe with gaze measurements during current
        # (i-th) light event
        flt4 = gaze4.loc[(gaze4.timeStamp>start4) & (gaze4.timeStamp<end4)]

        # Calculate distance from gaze point to light
        flt4["dst"] = np.sqrt(np.square(flt4.screenGazePosX-ox)+
                              np.square(flt4.screenGazePosY-oy))

        # Calculate shift of gaze in X and Y, from one event to the next
        flt4["dx"] = flt4['screenGazePosX'].diff()
        flt4["dy"] = flt4['screenGazePosY'].diff()

        # Save to Light Task dataframe the average distance of gaze points
        # to light point
        light4.dst[i]=flt4.dst.mean()

        # Save to Light Task dataframe the length of gaze path
        light4.lng[i]=np.sqrt((flt4.dx**2+flt4.dy**2)).sum()


flt4

# 5. Save All Light Task Events Filtered to a CSV

# Save new Light Task dataframe to new CSV file
# 1/8 ############################################################
light1.to_csv('S03_0037YF-L1_LightTask_Out.csv')
light2.to_csv('S03_0037YF-L2_LightTask_Out.csv')
light3.to_csv('S03_0037YF-L3_LightTask_Out.csv')
light4.to_csv('S03_0037YF-L4_LightTask_Out.csv')

# 6. Drop False Light Events from DataFrame

# 2/8 ############################################################

light1.dropna(inplace = True)
light1.to_csv('S03_0037YF-L1_LightTask_Out_noNA.csv')

light2.dropna(inplace = True)
light2.to_csv('S03_0037YF-L2_LightTask_Out_noNA.csv')

light3.dropna(inplace = True)
light3.to_csv('S03_0037YF-L3_LightTask_Out_noNA.csv')

light4.dropna(inplace = True)
light4.to_csv('S03_0037YF-L4_LightTask_Out_noNA.csv')


light4.head()

#plt.rcParams["figure.figsize"] = [7.50, 3.50]
#plt.rcParams["figure.autolayout"] = True

x1 = light1['timeDuration']
y1 = light1['dst']
z1 = light1['lng']

x2 = light2['timeDuration']
y2 = light2['dst']
z2 = light2['lng']

x3 = light3['timeDuration']
y3 = light3['dst']
z3 = light3['lng']

x4 = light4['timeDuration']
y4 = light4['dst']
z4 = light4['lng']

#plt.scatter(x, y, c ="blue")
# To show the plot
#plt.show()

# Initialise the subplot function using number of rows and columns
#figure, axis = plt.subplots(2, 2)

# 7. Plot Linear Regression of Distance vs. Light Task Duration (x)
```

```
fig, axes = plt.subplots(2, 2, sharex=False, sharey=True, figsize=(14, 10))
# 3/8 ##############################################################
fig.suptitle('Distance Gaze Center to Light vs. LightTask Duration\n Subject3',
             fontsize = 20)

fig.tight_layout(pad=3.5)
axes[0,0].set_title('Layout 1')
axes[0,1].set_title('Layout 2')
axes[1,0].set_title('Layout 3')
axes[1,1].set_title('Layout 4')

p1 = sns.regplot(ax=axes[0, 0], data=light1, x=x1, y=y1)
p2 = sns.regplot(ax=axes[0, 1], data=light2, x=x2, y=y2)
p3 = sns.regplot(ax=axes[1, 0], data=light3, x=x3, y=y3)
p4 = sns.regplot(ax=axes[1, 1], data=light4, x=x4, y=y4)

p1.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Distance Gaze Center to Light (px)")
p2.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Distance Gaze Center to Light (px)")
p3.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Distance Gaze Center to Light (px)")
p4.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Distance Gaze Center to Light (px)")


# 4/8 ##############################################################

plt.savefig('S03_0037YF-Combined_Dist_Dur.jpeg')


# 8. Calculate Linear Regression of Path Length (z) v. Light Task Duration (x)


fig, axes = plt.subplots(2, 2, sharex=False, sharey=True, figsize=(14, 10))

# 5/8 ##############################################################

fig.suptitle('Gaze Path Length vs. LightTask Duration\n Subject 3 0037YF',
             fontsize = 20)   #############
fig.tight_layout(pad=4.0)

axes[0,0].set_title('Layout 1')
axes[0,1].set_title('Layout 2')
axes[1,0].set_title('Layout 3')
axes[1,1].set_title('Layout 4')

j1 = sns.regplot(ax=axes[0, 0], data=light1, x=x1, y=z1)
j2 = sns.regplot(ax=axes[0, 1], data=light2, x=x2, y=z2)
j3 = sns.regplot(ax=axes[1, 0], data=light3, x=x3, y=z3)
```

```
j4 = sns.regplot(ax=axes[1, 1], data=light4, x=x4, y=z4)

j1.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Gaze Path Length to Light (px)")
j2.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Gaze Path Length to Light (px)")
j3.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Gaze Path Length to Light (px)")
j4.set(xlabel ="Duration of Light Task (ms)",
       ylabel = "Gaze Path Length to Light (px)")

# 6/8 ##############################################################

plt.savefig('S03_0037YF-Combined_Path_Dur.jpeg')

# 9. R Squared: Distance v. Duration of Light Task

#LAYOUT ONE

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light1[['timeDuration']]
y1 = light1[['dst']]

#fit regression model
model.fit(X, y1)

#calculate R-squared of regression model
r_squared_d1 = model.score(X, y1)

#view R-squared value
print(r_squared_d1)

#LAYOUT TWO

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light2[['timeDuration']]
y2 = light2[['dst']]

#fit regression model
model.fit(X, y2)

#calculate R-squared of regression model
r_squared_d2 = model.score(X, y2)
```

```
#view R-squared value
print(r_squared_d2)


#LAYOUT THREE

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light3[['timeDuration']]
y3 = light3[['dst']]

#fit regression model
model.fit(X, y3)

#calculate R-squared of regression model
r_squared_d3 = model.score(X, y3)

#view R-squared value
print(r_squared_d3)


#LAYOUT FOUR

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light4[['timeDuration']]
y4 = light4[['dst']]

#fit regression model
model.fit(X, y4)

#calculate R-squared of regression model
r_squared_d4 = model.score(X, y4)

#view R-squared value
print(r_squared_d4)


# 10. Save all Distance v. Dur R^2 Values to CSV


df_dr = np.array([r_squared_d1, r_squared_d2, r_squared_d3, r_squared_d4])

dist_rfin = pd.DataFrame(df_dr, columns = ['R Values'])
```

```
dist_rfin.to_csv('S02_002UG7-R2-Dist-Dur.csv')

print('R Squared: Path Length v. Duration of Light Task\n')
print('R Squared Layout 1 =', r_squared_d1)
print('R Squared Layout 2 =', r_squared_d2)
print('R Squared Layout 3 =', r_squared_d3)
print('R Squared Layout 4 =', r_squared_d4)


# 11. R Squared: Path Length v. Duration of Light Task

#LAYOUT ONE

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light1[['timeDuration']]
z1 = light1[['lng']]

#fit regression model
model.fit(X, z1)

#calculate R-squared of regression model
r_squared_p1 = model.score(X, z1)

#view R-squared value
print(r_squared_p1)

#LAYOUT TWO

#initiate linear regression model
model = LinearRegression()

#define predictor and response variables
X = light2[['timeDuration']]
z2 = light2[['lng']]

#fit regression model
model.fit(X, z2)

#calculate R-squared of regression model
r_squared_p2 = model.score(X, z2)

#view R-squared value
print(r_squared_p2)

#LAYOUT THREE

#initiate linear regression model
```

10.2 Appendix 2: 2D Processing Method

```
# Importing Libraries

import numpy as np
import pandas as pd
import numpy as np
import seaborn as sns

from matplotlib import pyplot as plt
from pandas.plotting import scatter_matrix
from sklearn.linear_model import LinearRegression

gaze = pd.read_csv('001RL5; 05-04-2021_16.15; EyeHeadTracking.txt')
light = pd.read_csv('001RL5; 05-04-2021_16.15; LightTask.txt')

gaze.describe()
axes = scatter_matrix(gaze, alpha=0.5, figsize=(50, 50), diagonal='hist')

# Define light position on screen, froom Top-Left of sctreen

lx=2217 # X coordinate of light point from Left of screen
ly=1181 # Y coordinate of light point from Top of screen

# Clean data ------------------------------------------------------------------

# Keep only Time stamp, duration and Gaze coordinates
gaze = gaze[['timeStamp', 'screenGazePosX', 'screenGazePosY']]

# Drop gaze points outside of screen
gaze.drop(gaze[(gaze.screenGazePosX>2880) |
               (gaze.screenGazePosX<0) |
               (gaze.screenGazePosY>1600) |
               (gaze.screenGazePosY<0)]
          .index, inplace=True)

##New # Clean Overlaps from light task data
# Drop incorrect light events ('lightCorrectness' == FALSE)
indexCorrect = light[ (light['lightCorrectness'] == False)].index
light.drop(indexCorrect , inplace=True)

# Drop light events that begin before previous event ends
indexOverlap = light[ light.timeStamp.diff() <
                      light.timeDuration.shift(1)].index
light.drop(indexOverlap , inplace=True)

light.reset_index(drop=True, inplace=True)

# For each light event in  LightTask file
# Calculate average distance from every gaze point to light point
# Calculate length of gaze path
# Calculate only for correct light events ('lightCorrectness' == TRUE)
```

```
# Add empty columns to light task dataframe, for avergae distance 'dst', and g
light[['dst', 'dst_s','dst_e','dst_d','lng','lng_rate']] = np.nan

# Add empty columns to gaze dataframe, for light task index
gaze[['light']] = np.nan
gaze['dx'] = gaze['screenGazePosX'].diff()
gaze['dy'] = gaze['screenGazePosY'].diff()
gaze['dt'] = gaze['timeStamp'].diff()
gaze['d_travel'] = np.sqrt((gaze.dx**2+gaze.dy**2))
gaze['speed'] = gaze.d_travel/gaze.dt
gaze = gaze[['timeStamp', 'screenGazePosX', 'screenGazePosY', 'd_travel', 'spe

for i in range(len(light.index)):
    if light.iloc[i]['lightCorrectness']: # only correct light events

        # start time stamp of gaze path, at start time stamp of light event
        start = light.iloc[i][0]

        # end time stamp of gaze path, at start time stamp of light event + Du
        end = light.iloc[i][0] + light.iloc[i][1]

        # set 'light' column of gaze to index of light task
        gaze['light'].loc[(gaze.timeStamp>start) & (gaze.timeStamp<end)] = i

        # Make temporary dataframe with gaze measurements during current (i-th
        flt = gaze.loc[(gaze.timeStamp>start) & (gaze.timeStamp<end)]

        # Calculate distance from gaze point to light
        flt["dst"] = np.sqrt(np.square(flt.screenGazePosX-lx)+
                             np.square(flt.screenGazePosY-ly))

        # Calculate shift of gaze in X and Y, from one event to the next
        flt["dx"] = flt['screenGazePosX'].diff()
        flt["dy"] = flt['screenGazePosY'].diff()

        # Save the average distance of gaze points to light point to Light Tas
        light.dst[i]=flt.dst.mean()
        light.dst_s[i]=flt.dst.iloc[0]
        light.dst_e[i]=flt.dst.iloc[-1]
        light.dst_d[i]=light.dst_e[i]-light.dst_s[i]

        # Save the length of gaze path to Light Task dataframe
        light.lng[i]=np.sqrt((flt.dx**2+flt.dy**2)).sum()
        light.lng_rate[i]=(np.sqrt((flt.dx**2+flt.dy**2)).sum())/
        light.timeStamp[i]

#End - Start NEGATIVE means the gaze moved towards the light

# -------- Movement Direction and Saccade/ Fixation --------------------------
```

```
light.loc[light['dst_d'] > 0, 'pos'] = 'Moved away from light'
light.loc[light['dst_d'] <= 0, 'pos'] = 'Moved towards light'

light.dropna(inplace=True)
light

#Defining a Saccade and Fixation
gaze.loc[gaze['speed'] > 0.2, 'Behavior'] = 'Saccade'
gaze.loc[gaze['speed'] <= 0.2, 'Behavior'] = 'Fixation'

# adding a light on column: binary, categorical.
# fill light n/a's first with something
gaze['light'] = gaze['light'].fillna(-1)
gaze.loc[gaze['light'] >= 0, 'LightStatus'] = 'ON'
gaze.loc[gaze['light'] < 0, 'LightStatus'] = 'OFF'

stdX = gaze.groupby(['light'])['screenGazePosX'].std()
stdY = gaze.groupby(['light'])['screenGazePosY'].std()

df_disp=pd.concat([stdX, stdY], axis = 1)

#renames distribution columns to he able to differentiate when appended
# to original dataframe
dispersions = df_disp.rename(columns = {"screenGazePosX" : "GazeDispersionX",
                                        "screenGazePosY" : "GazeDispersionY"})

features = pd.concat([light, dispersions], axis = 1)
features.head()

axes = scatter_matrix(features, alpha=1, figsize=(10, 8), diagonal='hist')
plt.show()

# Save new Light Task dataframe to new CSV file
light.to_csv('G_LightTask_StartEnd_S01_ff.csv')

# Save new Gaze dataframe to new CSV file
gaze.to_csv('G_Gaze_LightTaskIndex_S01_ff.csv')

# Save new concatenated Features dataframe to new CSV file
features.to_csv('G_Features_MacroLight_S01_ff.csv')

# -------- Data Exploration: Two Datasets -------------------------------

# Two Dataframes: [Features] and [Gaze]
# Gaze, Individual Movements - Descriptive Statistics
# TimeStamp, Screen Position, Screen Positon Y, Distance Travelled
# Speed (Distance travelled/ time), Light Event, Behavior, Light Status

# --- Gaze Events Grouped by Behavior, Light Status, and speed ---------
```

```
gaze_count = gaze.groupby(['Behavior','LightStatus'])['LightStatus'].count()
g_travelstats = gaze.groupby(['LightStatus', 'Behavior'])['d_travel']
gt_events = gaze.groupby(['LightStatus'])['d_travel']
gt_events = gaze.groupby(['LightStatus', 'light'])['d_travel']
gt_events = gaze.groupby(['LightStatus'])['speed']
gt_events = gaze.groupby(['LightStatus','light'])['speed']
gt_events = gaze.groupby(['LightStatus'])['speed'].hist()

# --- Data Visualization ------------------------------------------------

plt.figure(figsize = [28,16])
gfg = sns.scatterplot(data=gaze, x=gaze.screenGazePosX, y=gaze.screenGazePosY,
                      s=120)
gfg.set_ylim(0, 1600)
gfg.set_xlim(0, 2800)

plt.figure(figsize = [28,16])
gfg2 = sns.scatterplot(data=gaze, x=gaze.screenGazePosX, y=gaze.screenGazePosY
                       hue="light", palette="deep", s=125)
gfg2.set_ylim(0, 1600)
gfg2.set_xlim(0, 2800)

plt.figure(figsize = [28,16])
g1 = gaze[(gaze['light']>-1)]
gfg2 = sns.scatterplot(data=g1, x=g1.screenGazePosX, y=g1.screenGazePosY,
                       hue="light", palette="deep", s=150)
gfg2.set_ylim(0, 1600)
gfg2.set_xlim(0, 2800)
plt.xlabel('Screen Gaze X (pixels)', fontsize=28);
plt.ylabel('Screen Gaze Y (pixels)', fontsize=28);
plt.title('All Gaze During Light-On Events - Subject 1', fontsize=30)

plt.figure(figsize = [28,16])
g0 = gaze[(gaze['light']<0)]
gfg2 = sns.scatterplot(data=g0, x=g0.screenGazePosX, y=g0.screenGazePosY,
                       hue="light", palette="deep", s=150)
gfg2.set_ylim(0, 1600)
gfg2.set_xlim(0, 2800)
plt.xlabel('Screen Gaze X (pixels)', fontsize=28);
plt.ylabel('Screen Gaze Y (pixels)', fontsize=28);
plt.title('All Gaze Events, No Light Task - Subject 1', fontsize=30)

#relationship of age and income
sns.jointplot(x="screenGazePosX", y="screenGazePosY", data=gaze, kind ='hex')

#scatterplot for just one light
plt.figure(figsize = [28,16])
g2 = gaze[(gaze['light']==21.0)]
gfg2a = sns.scatterplot(data=gaze, x=g2.screenGazePosX, y=g2.screenGazePosY,
                        s=200)
```

```
gfg2a.set_ylim(0, 1600)
gfg2a.set_xlim(0, 2800)

plt.xlabel('Screen Gaze X (pixels)', fontsize=28);
plt.ylabel('Screen Gaze Y (pixels)', fontsize=28);
plt.title('All Gaze Events, Subject 1, Event 21', fontsize=30)

#scatterplot for just one light
plt.figure(figsize = [28,16])
g2 = gaze[(gaze['light']==21.0)]
gfg2b = sns.scatterplot(data=gaze, x=g2.screenGazePosX, y=g2.screenGazePosY,
                        hue=g2.timeStamp, s=200)
gfg2b.set_ylim(0, 1600)
gfg2b.set_xlim(0, 2800)

#scatterplot for just one light
plt.figure(figsize = [28,16])
g2 = gaze[(gaze['light']==21.0)]
gfg2c = sns.lineplot(data=gaze, x=g2.screenGazePosX, y=g2.screenGazePosY)
#gfg2a.set_ylim(0, 1600)
#gfg2a.set_xlim(0, 2800)

sns.scatterplot(data=g2, x="screenGazePosX", y="screenGazePosY",
                hue=g2.timeStamp)

gfg5a = sns.relplot(data=gaze, x="timeStamp", y="speed",
                    hue = "LightStatus")

#Distance Travelled During Light On/ Off Events
gfg5b = sns.relplot(data=gaze, x="timeStamp", y="d_travel",
                    hue = "LightStatus")

#Distance Travelled During ONE Light Event: #19
gfg5b = sns.relplot(data=g2, x="timeStamp", y="d_travel",
                    hue = "speed")

## IS THE SPEED FASTER AT THE BEGINNING OR THE END OF ONE EVENT ? ------------

#Distance Travelled During Light On/ Off Events
gfg5b = sns.relplot(data=g2, x="timeStamp", y="d_travel",hue = "Behavior")

# ARE THERE MORE FIXATIONS AT THE BEGINNING THAN AT THE END? ---------------

#Distance Travelled During Light On/ Off Events
gfg5b = sns.relplot(data=g2, x="timeStamp", y="speed",hue = "Behavior")

gfg5c = sns.relplot(
    data=gaze, x="timeStamp", y="d_travel",
    col="LightStatus", hue="Behavior", style="LightStatus",
    kind="scatter")
```

```
gaze.groupby('LightStatus')['speed'].plot(legend=True)
plt.xlabel("Measurement #", fontweight='bold')
ax = plt.axes()
ax.set_facecolor("white")

gfg7 = sns.relplot(
    data=gaze, x="timeStamp", y="speed",
    col="LightStatus", hue="Behavior", style="Behavior",
    kind="scatter"
)
gfg7.fig.suptitle("Behavior Across Time for Light on and Light Off Events")
gfg7.fig.tight_layout(pad=1.0)

gfg6 = sns.relplot(
    data=gaze, x="timeStamp", y="speed",
    col="Behavior", hue="LightStatus", style="Behavior",
    kind="scatter")

plt.figure(figsize = [100,40])
CID = gaze['light'].unique()
select = gaze.loc[gaze['light'].isin(CID)]
sns.violinplot('light', 'd_travel', data = select)
plt.grid()

gfg6 = sns.relplot(
    data=gaze, x="timeStamp", y="speed",
    col="LightStatus", hue="LightStatus", style="LightStatus",
    kind="scatter")

#Count of Fixations and Saccades in Light On SLOW vs. Light On FAST
### FOR THIS GO TO MACRO EVENT / FEATURES TABLE ---------------------------

# Features Dataset ----------------------------------------------------------
# - Index: Light Event Number, Distance, Start, End, Delta, Vector Away/From
# - Gaze Dispersion X , Gaze Dispersion Y

#Travel Distance Stats: Count, Max, Distribution, by Behavior and LightStatus
features.groupby(['pos'])['timeDuration'].count()
features.describe()
features.groupby(['pos'])['timeDuration'].describe()
features.groupby(['pos', 'dst_d'])['timeDuration'].describe()

#Fig3a. Distance between Start and End location During Light Events
gfg3a = sns.relplot(
    data=features, x="timeStamp", y="dst_d",
    col="pos", hue="timeDuration", style="pos",
    kind="scatter")

# BETWEEN THE GAZES THAT MOVED AWAY VS. MOVED TOWARDS THE LIGHT,
```

```
# WAS THERE A DIFFERENCE IN VARIANCE?

#Fig3a. Distance between Start and End location During Light Events
gfg3a = sns.relplot(
    data=features, x=features.index, y="timeDuration",
    col="pos", hue="pos", style="pos",
    kind="scatter")

#Fig3a. Distance between Start and End location During Light Events
gfg3a = sns.relplot(
    data=features, x=features.index, y="timeDuration",
    col="pos", hue="pos", style="pos",
    kind="line")

#Fig3a. Distance between Start and End location During Light Events
gfg3a = sns.relplot(
    data=features, x="GazeDispersionX", y="timeDuration",
    hue="pos", style="pos",
    kind="scatter")

#Fig3a. Distance between Start and End location During Light Events
GazeXY = features.GazeDispersionX + features.GazeDispersionY
gfg3a = sns.relplot(
    data=features, x="timeDuration", y=GazeXY,
    hue="pos", style="pos",
    kind="scatter")
gfg4 = sns.relplot(data=features, x="timeStamp", y="dst_d",hue = "pos")

#Fig3c. Gaze Dispersion During Events where movement was towards the light vs.
# away from the light
gfg3c = sns.relplot(
    data=features, x="timeStamp", y="GazeDispersionX",
    col="pos", hue="pos", style="pos",
    kind="scatter")

gfg3b = sns.relplot(
    data=features, x="dst_d", y="lng_rate", hue="pos", style="pos",
    kind="scatter")
```

10.3 Appendix 3: ML Models, 1 Subject

```
# # 1-2D Raw Gaze Annotated - KNN. RF. SVM. NBay. DT. PCA.
#ONE SUBJECT, NATIVE GAZE + DERVIED GAZE + APPENDED EVENT INDEX
#Importing libraries ------------------------------------------------------
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style("whitegrid")
#Importing ML models ------------------------------------------------------
#Libraries for data processing
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
#Libraries for decision tree and random forest model
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
#For performing hyperparameter tuning
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import GridSearchCV
# Regression ---------------------------------------------------------------
from sklearn.linear_model import LogisticRegression
from scipy.special import expit
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import RFECV
import matplotlib.mlab as mlab
import scipy.stats as st
import math
#PCA
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
#KNN
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
#SVM
from sklearn import svm
import warnings
warnings.filterwarnings('ignore')
#Evaluation metrics
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
, recall_score, f1_score
from sklearn.metrics import classification_report
from sklearn.linear_model import*
from sklearn.ensemble import*
from sklearn import metrics
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import KFold
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve
# ------------------------------------------------------------------------\
```

```
# IMPORT DATA: ONE SUBJECT, NATIVE GAZE + DERVIED GAZE + APPENDED EVENT INDEX
gaze = pd.read_csv('G_Gaze_LightTaskIndex_S01_ff.csv')
gaze = gaze.drop(['Unnamed: 0'], axis=1)
gaze.head()
gaze.dtypes
gaze.dropna(inplace=True)

#finding categorical data in gaze: ----------------------------------------
ccol = []
for column in gaze.columns:
    if gaze[column].dtype == object and len(gaze[column].unique()) <= 50:
        ccol.append(column)
        print(f"{column} : {gaze[column].unique()}")

#label encoding the above
label = LabelEncoder()
for column in ccol:
    gaze[column] = label.fit_transform(gaze[column])
biglights.dtypes
# SET X FEATURES AND Y TARGET  ---------------------------------------------
X = gaze[['speed','screenGazePosX', 'screenGazePosY', 'Behavior', 'd_travel']]
y = gaze['LightStatus']

# SSET 30% OF DATA TO TEST AND 70% TO TRAINING OF MODEL  -------------------
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=0)

# KNN ----------------------------------------------------------------------
# First learning model (k = 3)
classifier = KNeighborsClassifier(n_neighbors=3)
# Fitting the model
classifier.fit(X_train, y_train)
# Predicting the Test set results
y_pred = classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)

## Cross Validation
from sklearn.model_selection import cross_val_score
# Trying different K values from 1 till 30 with step by 2
# Choosing the best K value for the given model based on the accuracy obtained
k_list = list(range(1,30,2))
# Creating list of Average Accuracy for each Cross-validation
cv_scores = []
# Performing 7-fold cross validation
for k in k_list:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X_train, y_train, cv=7, scoring='accuracy')
    cv_scores.append(scores.mean())
```

```
#plot the neighbor selection graph to select the best number of neighbors: k
plt.figure()
#plt.figure(figsize=(15,10))
plt.title('The optimal number of neighbors', fontsize=20, fontweight='bold')
plt.xlabel('Number of Neighbors K', fontsize=15)
plt.ylabel('Accuracy', fontsize=15)
sns.set_style("whitegrid")

plt.plot(k_list, cv_scores)
plt.show()

# Use knn model and the best k value: 7, to report the accuracy of the new model
classifier = KNeighborsClassifier(n_neighbors=16)
classifier.fit(X_train, y_train)
y_pred5 = classifier.predict(X_test)
accuracy_score(y_pred5,y_test)

# Random Forest ------------------------------------------------------------
clf=RandomForestRegressor(random_state=0)
clf.fit(X_train,y_train)
y_tr1=clf.predict(X_train)
y_pr=clf.predict(X_test)

print('train data accuracy :',clf.score(X_train,y_train))
print('test data accuracy :',clf.score(X_test,y_test))
print('loss of train data :',mean_squared_error(y_train,y_tr1))
print('loss of test data :',mean_squared_error(y_test,y_pr))

# SVM ----------------------------------------------------------------------
clf = svm.SVC()
clf.fit(X_train, y_train) #train model
y_preds = clf.predict(X_test) #get prediction
#Evaluating our model
print(confusion_matrix(y_test,y_preds))
print(classification_report(y_test,y_preds))

# Decision Tree ------------------------------------------------------------
#train model
tree_clf = DecisionTreeClassifier(random_state=42)
tree_clf.fit(X_train, y_train)
#create function to get accuracy and confusion matrix
def print_score(clf, X_train, y_train, X_test, y_test, train=True,
                pos_label="Yes"):
    if train == False:
        pred = clf.predict(X_test)
        print("Test Result:\n")
        print(f"accuracy score: {accuracy_score(y_test, pred)}\n")
        print(f"Confusion Matrix: \n {confusion_matrix(y_test, pred)}\n")

#report the result
```

```
print_score(tree_clf, X_train, y_train, X_test, y_test, train=False)

pred = tree_clf.predict(X_test)
cm_n = confusion_matrix(y_test,pred)
conf_matrix_n =pd.DataFrame(data=cm_n,columns=['Predicted:0','Predicted:1'],
                            index=['Actual:0','Actual:1'])
plt.figure(figsize = (8,5))
sns.heatmap(conf_matrix_n, annot=True, fmt='d', cmap="YlGnBu")
plt.figure(figsize = (12,8))
plt.show()

# Naive bayes --------------------------------------------------------------

#train model and make prediction
from sklearn.naive_bayes import GaussianNB
naive_bayes = GaussianNB()
naive_bayes.fit(X_train,y_train)
y_predn = naive_bayes.predict(X_test)

#report the result
print(accuracy_score(y_predn,y_test))
cm = confusion_matrix(y_test,y_predn)
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:1'],
                         index=['Actual:0','Actual:1'])
plt.figure(figsize = (8,5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap="YlGnBu")
plt.figure(figsize = (12,8))
plt.show()

# PCA # --------------------------------------------------------------------
#use pca to reduce dimensions
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca_result = pca.fit_transform(X)
pca.explained_variance_ratio_
X1 = pca_result
y1 = y.copy()
y1 = y1.to_numpy()
NB = GaussianNB()
NB.fit(X1,y1)
pca_result

l, r = X1[:, 0].min() - 1, X1[:, 0].max() + 1
b, t = X1[:, 1].min() - 1, X1[:, 1].max() + 1
n = 200
grid_x, grid_y = np.meshgrid(np.linspace(l, r, n), np.linspace(b, t, n))
test_x = np.column_stack((grid_x.ravel(), grid_y.ravel()))
y_ = NB.predict(test_x)
grid_z = y_.reshape(grid_x.shape)
plt.pcolormesh(grid_x, grid_y, grid_z, cmap=plt.cm.Pastel1)
```

```
plt.scatter(X1[:,0],X1[:,1],c=y1,cmap=plt.cm.Oranges,s=50,marker="*")
plt.xlabel("principal component 1")
plt.ylabel("principal component 2")
plt.title("Use Gaussion naive bayes model")
plt.show()
y1_pred3 = NB.predict(X1)
accuracy_score(y1,y1_pred3)
```

10.4 Appendix 4: Logistic Regression Method

```
## LOGISTIC REGRESSION FUNCTION ## ----------------------------------------
#Importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
#Libraries for data processing
from sklearn import metrics
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
#Evaluation metrics
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
    recall_score, f1_score, mean_squared_error, r2_score, roc_curve
from sklearn.linear_model import*
from sklearn.ensemble import*
from sklearn.model_selection import KFold
#Importing Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from scipy.special import expit
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import RFECV
import matplotlib.mlab as mlab
import scipy.stats as st
import math
import warnings

get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style("whitegrid")
warnings.filterwarnings('ignore')

# ----------------------------------------------------------------------
gaze = pd.read_csv('G_Gaze_LightTaskIndex_S01_ff.csv')
# Save new concatenated Features dataframe to new CSV file
biglights = pd.read_csv('G_Features_MacroLight_S01_ff.csv')
gaze = gaze.drop(['Unnamed: 0'], axis=1)
biglights = biglights.drop(['Unnamed: 0','lightCorrectness'], axis =1)

# ----------------------------------------------------------------------
gaze.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20))
biglights.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20))
gaze.dropna(inplace=True)
# No NULL values
gaze.isnull().sum()
# ----------------------------------------------------------------------
plt.figure(figsize = (12,8))
corrplot = sns.heatmap(gaze.corr().round(2), annot=True)
corrplot.set_xticklabels(corrplot.get_xticklabels(), rotation = 45,
                         horizontalalignment = 'right') #For X axis labels
# ----------------------------------------------------------------------
```

```
#finding categorical data in gaze #ccol = categorical columns
ccol = []
for column in gaze.columns:
    if gaze[column].dtype == object and len(gaze[column].unique()) <= 50:
        ccol.append(column)
        print(f"{column} : {gaze[column].unique()}")
# ----------------------------------------------------------------------
#label encoding the above
label = LabelEncoder()
for column in ccol:
    gaze[column] = label.fit_transform(gaze[column])
# ----------------------------------------------------------------------
biglights.dropna(inplace=True)
biglights.isnull().sum()
biglights['pos']
# ----------------------------------------------------------------------
#finding categorical data in the dg
ccol_BL = []
for column in biglights.columns:
    if biglights[column].dtype == object and
        len(biglights[column].unique()) <= 50:
        ccol_BL.append(column)
        print(f"{column} : {biglights[column].unique()}")
#label encoding the above
label = LabelEncoder()
for column in ccol_BL:
    biglights[column] = label.fit_transform(biglights[column])
#High level correlation
plt.figure(figsize = (12,8))
corrplot2 = sns.heatmap(biglights.corr().round(2), annot = True)
# ----------------------------------------------------------------------
X = biglights[['dst','dst_d','lng','lng_rate','GazeDispersionX',
               'GazeDispersionY']]
y = biglights['timeDuration']
pcorr_matrix = X.corr().round(2) #High level correlation heat map
plt.figure(figsize = (8,5))
sns.heatmap(pcorr_matrix, annot = True) #True is used to print values
# ----------------------------------------------------------------------
#Splitting the data into training and testing (70:30)
X1 = biglights[['lng']]
Y1 = biglights['pos']
X_train, X_test, y_train, y_test = train_test_split(X1, Y1, test_size=0.3,
                                                    random_state=0)

logreg = LogisticRegression() #Instantiating the LogisticRegression Object
logreg.fit(X_train, y_train) #Fitting the model on our training data
y_pred = logreg.predict(X_test) #Making predictions on Testing Model
#Creating Confusion Matrix and display acc
cm = confusion_matrix(y_test,y_pred)
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:1'],
```

```
                          index=['Actual:0','Actual:1'])
plt.figure(figsize = (8,5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap="YlGnBu")
# ------------------------------------------------------------------------------
#True Negative, True Positive, False Negative, False Positive
TN=cm[0,0]
TP=cm[1,1]
FN=cm[1,0]
FP=cm[0,1]
#Accuracy
print('Accuracy  = TP+TN/(TP+TN+FP+FN) = ',
      round((TP+TN)/float(TP+TN+FP+FN),3))
# ------------------------------------------------------------------------------
#get intercept and coeficient of the logreg model
model = LogisticRegression(random_state=0).fit(X1, Y1)
print("Logistic Model Intercept:", model.intercept_)
print("Logistic Model Coefficient:", model.coef_)

#set sigmoid function and plot it
sigmoid_function = expit(X1 * model.coef_ + model.intercept_)
plt.scatter(X1, sigmoid_function)
plt.title('Sigmoid curve')
plt.xlabel('Thresholds')
plt.ylabel('Classification value')

#Find the reasonable decision threshold number
decision_thresh = (math.log((1 / 0.5) - 1) - model.intercept_) / model.coef_
print("Decision threshold:", decision_thresh)

# LOGISTIC Regression 5: DIGITIZE DURATION @ 2000 #-------------------------

import statsmodels.api as sm
logit_model=sm.Logit(Y2,X2)
result=logit_model.fit()
print(result.summary())

a = biglights['timeDuration']
bins = np.array([2000])
gfg = np.digitize(a, bins)

#Splitting the data into training and testing (70:30)
X2 = biglights[['GazeDispersionX']]
Y2 = gfg
X_train2, X_test2, y_train2, y_test2 = train_test_split(X2, Y2, test_size=0.3,
                                                        random_state=0)
#------------------------------------------------------------------------------
logreg = LogisticRegression() #Instantiating the LogisticRegression Object
logreg.fit(X_train2, y_train2) #Fitting model on training data w/ fit method
y_pred2 = logreg.predict(X_test2) #Making predictions on Testing Model
#------------------------------------------------------------------------------
```

```
#Creating Confusion Matrix and display acc
cm = confusion_matrix(y_test2,y_pred2)
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:1'],
                         index=['Actual:0','Actual:1'])
plt.figure(figsize = (8,5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap="YlGnBu")

#True Negative, True Positive, False Negative, False Positive
TN=cm[0,0]
TP=cm[1,1]
FN=cm[1,0]
FP=cm[0,1]

print('Accuracy  = TP+TN/(TP+TN+FP+FN) = ',
      round((TP+TN)/float(TP+TN+FP+FN),3))
#get intercept and coeficient of the logreg model
model = LogisticRegression(random_state=0).fit(X2, Y2)
print("Logistic Model Intercept:", model.intercept_)
print("Logistic Model Coefficient:", model.coef_)
#------------------------------------------------------------------------------
#set sigmoid function and plot it
sigmoid_function = expit(X2 * model.coef_ + model.intercept_)
plt.scatter(X2, sigmoid_function)
plt.title('Response Time vs. Gaze Dispersion X')
plt.xlabel('Gaze Dispersion, X')
plt.ylabel('Time Duration, Binned')
```

10.5 Appendix 5: Decision Tree Method

```
#Importing libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style("whitegrid")

#Libraries for data processing
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

#Libraries for decision tree and random forest model
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split

#For performing hyperparameter tuning ( advanced code)
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import GridSearchCV

#Evaluation metrics
from sklearn.metrics import accuracy_score, precision_score
from sklearn.metrics import recall_score, f1_score
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn.metrics import r2_score
from sklearn.model_selection import KFold
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve

from scipy.special import expit
from sklearn.metrics import roc_curve
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import RFECV
import matplotlib.mlab as mlab
import scipy.stats as st
import math
import warnings
warnings.filterwarnings('ignore')

# # From Gaze Dataset: Classifying Gaze Behavior
gaze = pd.read_csv('G_Gaze_LightTaskIndex_S01_ff.csv')

# Save new concatenated Features dataframe to new CSV file
biglights = pd.read_csv('G_Features_MacroLight_S01_ff.csv')
```

```
gaze = gaze.drop(['Unnamed: 0'], axis=1)
biglights = biglights.drop(['Unnamed: 0','lightCorrectness'], axis =1)

gaze.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20))
plt.show()

biglights.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20))

gaze.dtypes
gaze.dropna(inplace=True)

# No NULL values
gaze.isnull().sum()
gaze.corr()

plt.figure(figsize = (12,8))
corrplot = sns.heatmap(gaze.corr().round(2), annot=True)
corrplot.set_xticklabels(corrplot.get_xticklabels(), rotation = 45, horizontal
plt.show()
#OBSERVATION: The further you go, the faster you go

#finding categorical data in gaze #ccol = categorical columns
ccol = []
for column in gaze.columns:
    if gaze[column].dtype == object and len(gaze[column].unique()) <= 50:
        ccol.append(column)
        print(f"{column} : {gaze[column].unique()}")

#label encoding the above
label = LabelEncoder()
for column in ccol:
    gaze[column] = label.fit_transform(gaze[column])

biglights.dtypes

biglights.shape
biglights.dropna(inplace=True)
biglights.isnull().sum()
biglights['pos']

#finding categorical data in the dg
ccol_BL = []
for column in biglights.columns:
    if biglights[column].dtype == object and len(biglights[column].unique())
    <= 50:
        ccol_BL.append(column)
        print(f"{column} : {biglights[column].unique()}")

#label encoding the above
label = LabelEncoder()
```

```
for column in ccol_BL:
    biglights[column] = label.fit_transform(biglights[column])

biglights.count()
biglights.corr()

#High level correlation
plt.figure(figsize = (12,8))
corrplot2 = sns.heatmap(biglights.corr().round(2), annot = True)
corrplot2

# DECISION TREE ------------------------------------------------------------

# # Selecting the Depedent and Independent Variables

X = biglights[['dst','dst_d','lng','lng_rate','GazeDispersionX',
               'GazeDispersionY']]
Y = biglights['pos']

X.columns

#High level heat map to display correlation
pcorr_matrix = X.corr().round(2)
#heatmap plotting
plt.figure(figsize = (8,5))
sns.heatmap(pcorr_matrix, annot = True) #Annot: used to print inside square
plt.show()


# ## Split the Dataset Into Train and Test Sets

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3)

# ## Fit The Decision Tree Classifier to the Training Data Set

clf = DecisionTreeClassifier(max_depth=4)
clf = clf.fit(X_train, y_train)

pp = pd.DataFrame([clf.get_params()])
pp.T

predictions = clf.predict(X_test)
predictions

clf.predict_proba(X_test)

##Evaluation

print(f"accuracy score: {accuracy_score(y_test, predictions)}\n")
print(f"Precision score: {precision_score(y_test,predictions)}\n")
```

```
print(f"Recall score: {recall_score(y_test, predictions)}\n")
print(f"Confusion Matrix: \n {confusion_matrix(y_test, predictions)}\n")

# -------- Built-in Visualization for Decision tree ------------------------

fn = X.columns
cn=['1','0']


from sklearn import tree

fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (8,8), dpi=300)
tree.plot_tree(clf,
               feature_names = fn,
               class_names=cn,
               filled = True,proportion=True, fontsize=10);
fig.savefig('DT_1.png')

# Visualize Feature Importance  ---------------------------------------------

clf.feature_importances_

feature_importance = pd.DataFrame(clf.feature_importances_,index = fn)
feature_importance

feature_importance.plot(kind='bar')

# D-TREE VIZ Feature VIsualizer ---------------------------------------------

from dtreeviz.trees import *
import dtreeviz

clfdt = DecisionTreeClassifier(max_depth=4)
clfdt.fit(X, Y)
viz_model = dtreeviz.model(clfdt, X, Y, feature_names=fn,target_name='pos',
                           class_names=cn)

v = viz_model.view()     # render as SVG into internal object
from graphviz import Source
import graphviz
```

10.6 Appendix 5: Automation Method

```
import numpy as np
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.linear_model import LinearRegression
import warnings
warnings.filterwarnings('ignore')
data = pd.read_csv("SubjectLayout.csv")
df = pd.DataFrame(data)
pre_f = '000-NavOut_'
pre_e = '000-GazeOut_'
Name_f = '_Filter_Clean-3sec'
Name_e = '_Gaze_BackFill-3sec'
Angles = '_AnglesMerged-3sec'
ML = '_MLOut-3sec'
bl = '_'
event_timeinterval = 3000

for s in range(df.shape[0]): #Iterate through subject names from file
    subj = df.iloc[s,0]
    for ly in range(df.shape[1]-1): #Iterates through layouts from file
        layout = df.iloc[s,ly+1] #ly is 0 - 4, in column 1-5
        # input file
        gaze_fn = subj + layout + '; EyeHeadTracking.txt'
        nav_fn = subj + layout + '; NavTools.txt'
        gaze = pd.read_csv(gaze_fn)
        nav = pd.read_csv(nav_fn)
        # Data Cleaning - Drop gaze points outside of screen
        gaze.drop(gaze[(gaze.screenGazePosX>2880) |
               (gaze.screenGazePosX<0) |
               (gaze.screenGazePosY>1600) |
               (gaze.screenGazePosY<0) |
               (gaze.screenGazePosZ < 0.02)].index, inplace=True)

        # Data Cleaning - Drop Gaze Extra Columns
        gaze.drop(['headRotQuatX', 'headRotQuatY', 'headRotQuatZ',
                   'headRotQuatW'], axis=1, inplace = True)

        #Data Cleaning - Drop NavEvents TurnOff and UserError Events
        nav.drop(nav[nav['extraInfo'] == 'TURNOFF'].index, inplace=True)
        nav.drop(nav[nav['extraInfo'] == 'USERERROR'].index, inplace=True)

        # Define the output file name based on formula
        filter_file_out = f"{pre_f}{subj}{bl}{layout}{Name_f}.csv"
        gaze_file_out = f"{pre_e}{subj}{bl}{layout}{Name_e}.csv"
        angles_file_out = f"{pre_e}{subj}{bl}{layout}{Angles}.csv"
        ML_file_out = f"{pre_e}{subj}{bl}{layout}{ML}.csv"

        #Define 3D Gaze Metrics:

        #Select head position and world gaze position from Eye file
        xo=gaze.headPosX
        zo=gaze.headPosZ
        wxo=gaze.worldGazePosX
        wzo=gaze.worldGazePosZ
        wyo=gaze.worldGazePosY

        #Create Gaze and Head Vectors
        ts = gaze.timeStamp
        normgz = np.sqrt(np.square(gaze.worldGazePosX-gaze.headPosX)
                 +np.square(gaze.worldGazePosZ-gaze.headPosZ))
        gazex_full = gaze.wGX - gaze.headPosX
        gazez_full = gaze.wGZ - gaze.headPosZ
        gazex = (gaze.wGX - gaze.headPosX)/ normgz
        gazez = (gaze.wGZ - gaze.headPosZ)/ normgz
        # Head X and Head Z Angles. used to project vector visually.
        headx = -np.sin(-np.deg2rad(gaze.headRotEulerY))
        headz = -np.cos(-np.deg2rad(gaze.headRotEulerY))

        # Change in Head Rotation Angles
        headRotEulerY_delta = gaze.headRotEulerY.diff()
        headRotEulerY_delta[abs(headRotEulerY_delta)>180] =
        headRotEulerY_delta-360*np.sign(headRotEulerY_delta)
        posheadx_d = gaze.headPosX.diff()
        posheadz_d = gaze.headPosZ.diff()
        posheady_d = gaze.headPosY.diff()
        path_length = np.sqrt(np.square(posheadx_d)+np.square(posheadz_d))
        path_length3D = np.sqrt(np.square(posheadx_d)+np.square(posheadz_d)+
                                np.square(posheady_d))
        pathx = posheadx_d/path_length
        pathz = posheadz_d/path_length
        #headRotEulerY_delta.plot()

        #Calculate Angle from Path to Head
        #dot product
        dotpathhd = pathx * headx + pathz * headz
        #cross product
        crosspathhd = pathx * headz - pathz * headx
        ang_pathheadc = np.degrees(np.arccos(dotpathhd))
        ang_pathheads = np.degrees(np.arcsin(crosspathhd))

        #Calculate Angle from Gaze to Head
        # dot product
        dotgzhd = gazex * headx + gazez * headz
        #cross product
        crossgzhd = gazex * headz - gazez * headx
        ang_gazeheadc = np.degrees(np.arccos(dotgzhd))
        ang_gazeheads = np.degrees(np.arcsin(crossgzhd))
```

```
timeint = 0.001 * gaze.timeStamp.diff()
#speed_head = df['headPosX'].diff()/timeint
speed_gazeheadc = ang_gazeheadc.diff()/timeint
speed_gazeheads = ang_gazeheads.diff()/timeint
speed_pathheadc = ang_pathheadc.diff()/timeint
speed_pathheads = ang_pathheads.diff()/timeint
#headRotEulerY_delta = df.headRotEulerY.diff()
speed_HeadRotEulerY = headRotEulerY_delta/timeint

#How fast they were Walking (head position)
#Speed of Walking (head position)
angles_o = pd.DataFrame({
    'Subject' : subj,
    'Layout' : layout,
    'timeStamp': ts,
    'ChangeHeadRotation': headRotEulerY_delta,
    'Speed_HeadRotEulerY': speed_HeadRotEulerY,
    'AngleGazetoHead': ang_gazeheadc,   #arcosine
    'AngleSpeed_GzHd': speed_gazeheadc, #arcosine
})
angles_o = angles_o.iloc[2:]
ang_merge = gaze.merge(angles_o, on='timeStamp')
ang_merge.to_csv(angles_file_out, index=False)
#### Extraction Function ######
ang = ang_merge
# Create Empty Columns for All New Calculated Values
nav[['Subject','Layout','TotPathLength_B', 'TotPathLength_A',
     'Ang_GzHd_B', 'Ang_GzHd_A','AngSpd_GzHd_B',
     'AngSpd_GzHd_A', 'RotChange_B', 'RotChange_A','HdRotEulerY_B',
     'HdRotEulerY_A',
     'avPupDiam_B', 'avPupDiam_A','avgOpen_B', 'avgOpen_A',
     'GazeSpeed_B', 'GazeSpeed_A', 'WalkSpeed_B',
     'WalkSpeed_A']] = np.nan
gaze[['event']] = -1 ##New replaced NANs with -1
gaze[['class']] = np.nan
# create columns on native file for head orientation
gaze[['HeadDir_delta']] = headRotEulerY_delta
gaze[['Speed_HdDir_Delta']] = speed_HeadRotEulerY
# create columns on native file for angle between gaze and head Y
gaze[['Ang_GzHd']] = abs(ang_gazeheads)
# create columns on native file for step length and gaze length in 3D
gaze[['Gaze_PathLength']] = np.sqrt(wxo.diff()**2+
                                    wyo.diff()**2+wzo.diff()**2)
gaze[['Step_Length']] = path_length3D
# create columns on native file for step speed and gaze speed in 3D
gaze[['Gaze_Speed']] = gaze.Gaze_PathLength/timeint
gaze[['Walk_Speed']] = path_length3D/timeint

b = 'before'
a = 'after'
```

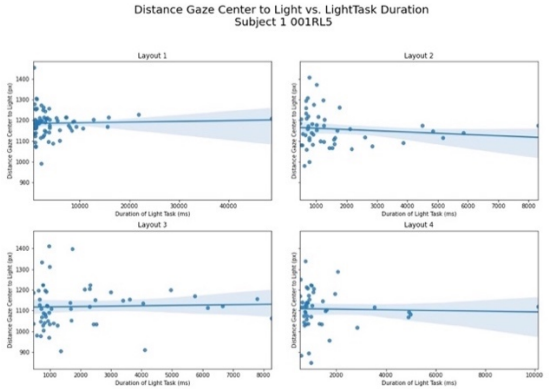
```
for i in range(len(nav.index)):
    start_b = nav.iloc[i][0] - event_timeinterval
    end_o= nav.iloc[i][0]
    end_a= nav.iloc[i][0] + event_timeinterval

    gaze['event'].loc[(gaze.timeStamp>start_b) &
                      (gaze.timeStamp<end_o)] = i
    gaze['event'].loc[(gaze.timeStamp>end_o) &
                      (gaze.timeStamp<end_a)] = i
    gaze['class'].loc[(gaze.timeStamp>start_b) &
                      (gaze.timeStamp<end_o)] = b
    gaze['class'].loc[(gaze.timeStamp>end_o) &
                      (gaze.timeStamp<end_a)] = a
    gaze['Subject'] = subj
    gaze['Layout'] = layout
    nav['Subject'] = subj
    nav['Layout'] = layout
    flt1_before = gaze.loc[(gaze.timeStamp>start_b) &
                           (gaze.timeStamp<end_o)]
    flt2_after = gaze.loc[(gaze.timeStamp>end_o) &
                          (gaze.timeStamp<end_a)]
    flta_before = ang.loc[(ang.timeStamp>start_b) &
                          (ang.timeStamp<end_o)]
    flta_after = ang.loc[(ang.timeStamp>end_o) &
                         (ang.timeStamp<end_a)]

    # Calculate shift of gaze in X and Y, from one event to the next
    flt1_before["dx"] = flt1_before['worldGazePosX'].diff()
    flt1_before["dy"] = flt1_before['worldGazePosY'].diff()
    flt1_before["dz"] = flt1_before['worldGazePosZ'].diff()
    flt2_after["dx"] = flt2_after['worldGazePosX'].diff()
    flt2_after["dy"] = flt2_after['worldGazePosY'].diff()
    flt2_after["dz"] = flt2_after['worldGazePosZ'].diff()

    # Calculate shift of head position in X and Y between events
    flt1_before["hpx"] = flt1_before['headPosX'].diff()
    flt1_before["hpz"] = flt1_before['headPosZ'].diff()
    flt2_after["hpx"] = flt2_after['headPosX'].diff()
    flt2_after["hpz"] = flt2_after['headPosZ'].diff()
    # Save length of 3D world gaze path to Nav dataframe

    nav.TotPathLength_B[i]=np.sqrt((flt1_before.dx**2+
                        flt1_before.dy**2+flt1_before.dz**2)).sum()
    nav.TotPathLength_A[i]=np.sqrt((flt2_after.dx**2+
                        flt2_after.dy**2+flt2_after.dz**2)).sum()

    nav.Ang_GzHd_B[i]= flta_before.AngleGazetoHead.std()
    nav.Ang_GzHd_A[i]= flta_after.AngleGazetoHead.std()

    nav.AngSpd_GzHd_B[i]= flta_before.Speed_HeadRotEulerY.mean()
```

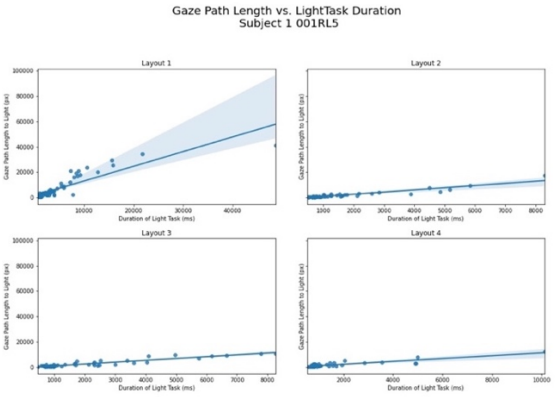
```
nav.AngSpd_GzHd_A[i]= flta_after.Speed_HeadRotEulerY.mean()

nav.RotChange_B[i]= flta_before.ChangeHeadRotation.std()
nav.RotChange_A[i]= flta_after.ChangeHeadRotation.std()

nav.HdRotEulerY_B[i]=flta_before.headRotEulerY.mean()
nav.HdRotEulerY_A[i]=flta_after.headRotEulerY.mean()

nav.GazeSpeed_B[i]=(np.sqrt((flt1_before.dx**2+flt1_before.dy**2+
        flt1_before.dz**2)).sum())/(event_timeinterval*0.001)
nav.GazeSpeed_A[i]=(np.sqrt((flt2_after.dx**2+flt2_after.dy**2+
            flt2_after.dz**2)).sum())/(event_timeinterval*0.001)

nav.WalkSpeed_B[i]=(np.sqrt((flt1_before.hpx**2+
        flt1_before.hpz**2)).sum())/(event_timeinterval*0.001)
nav.WalkSpeed_A[i]=(np.sqrt((flt2_after.hpx**2+
            flt2_after.hpz**2)).sum())/(event_timeinterval*0.001)

nav.avPupDiam_B[i] = flta_before.avgPupilDiameter.mean()
nav.avPupDiam_A[i] = flta_after.avgPupilDiameter.mean()

nav.avgOpen_B[i] = flta_before.avgOpenness.mean()
nav.avgOpen_A[i] = flta_after.avgOpenness.mean()

#Calculate Gaze Dispersion from events and set to a new dataframe
stdX = gaze.groupby(['event'])['worldGazePosX'].std()
stdY = gaze.groupby(['event'])['worldGazePosY'].std()
stdZ = gaze.groupby(['event'])['worldGazePosZ'].std()

df_disp=pd.concat([stdX, stdZ], axis = 1)
dispersions = df_disp.rename(columns = {"worldGazePosX" :
        "GazeDispersionX", "worldGazePosZ" : "GazeDispersionZ"})

features = pd.concat([nav, dispersions], axis = 1)

nav2 = features.dropna() #data cleaning - drop NA's
#nav = nav.reset_index()
# Transpose Aggregate Layout for ML
# Make NEW dataframe with rows from OLD, repeated 2 times
new = pd.DataFrame(np.repeat(nav2.values, 2, axis=0))

# Rename NEW Dataframe columns like OLD columns
new.columns = nav2.columns

# Make new column for WHEN
new[['When']] = np.nan

# Set WHEN to "before" for even rows, to "after" for odd rows
new.loc[::2,'When'] = "before"
new.loc[1::2,'When'] = "after"
```

```
    # Make new column DATA for combined before and after data
    new[['TotPathLength']] = np.nan
    new[['Ang_GzHd']] = np.nan
    new[['AngSpd_GzHd']] = np.nan
    new[['RotChange']] = np.nan
    new[['HdRotEulerY']] = np.nan
    new[['avPupDiam']] = np.nan
    new[['avgOpen']] = np.nan
    new[['WalkSpeed']] = np.nan
    # Set DATA to old.before for even rows, to old.after for odd rows
    new.loc[::2,'TotPathLength'] = new.loc[::2,'TotPathLength_B']
    new.loc[1::2,'TotPathLength'] = new.loc[1::2,'TotPathLength_A']
    new.loc[::2,'Ang_GzHd'] = new.loc[::2,'Ang_GzHd_B']
    new.loc[1::2,'Ang_GzHd'] = new.loc[1::2,'Ang_GzHd_A']
    new.loc[::2,'AngSpd_GzHd'] = new.loc[::2,'AngSpd_GzHd_B']
    new.loc[1::2,'AngSpd_GzHd'] = new.loc[1::2,'AngSpd_GzHd_A']
    new.loc[::2,'RotChange'] = new.loc[::2,'RotChange_B']
    new.loc[1::2,'RotChange'] = new.loc[1::2,'RotChange_A']
    new.loc[::2,'HdRotEulerY'] = new.loc[::2,'HdRotEulerY_B']
    new.loc[1::2,'HdRotEulerY'] = new.loc[1::2,'HdRotEulerY_A']
    new.loc[::2,'avPupDiam'] = new.loc[::2,'avPupDiam_B']
    new.loc[1::2,'avPupDiam'] = new.loc[1::2,'avPupDiam_A']
    new.loc[::2,'avgOpen'] = new.loc[::2,'avgOpen_B']
    new.loc[1::2,'avgOpen'] = new.loc[1::2,'avgOpen_A']
    new.loc[::2,'WalkSpeed'] = new.loc[::2,'WalkSpeed_B']
    new.loc[1::2,'WalkSpeed'] = new.loc[1::2,'WalkSpeed_A']
    # # Drop original columns 'Bef_Data', 'Aft_Data'
    new.drop(['TotPathLength_B', 'TotPathLength_A',
              'Ang_GzHd_B', 'Ang_GzHd_A',
              'AngSpd_GzHd_B', 'AngSpd_GzHd_A', 'RotChange_B',
              'RotChange_A',
              'HdRotEulerY_B', 'HdRotEulerY_A', 'avPupDiam_B' ,
              'avPupDiam_A',
              'avgOpen_B','avgOpen_A', 'WalkSpeed_B','WalkSpeed_A'],
             axis=1, inplace=True)
# Final Export of FIles
nav2.to_csv(filter_file_out, index=False) #Nav filtered with calculate
gaze.to_csv(gaze_file_out, index=False) #Gaze file with back-propagate
new.to_csv(ML_file_out, index=False) #transposed format with Before/Af
```

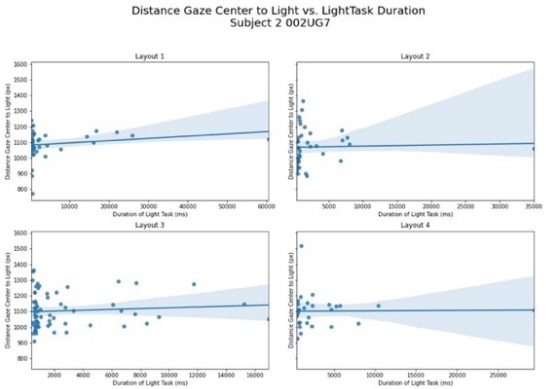
10.7 Appendix 6: Baseline Trend Visualization

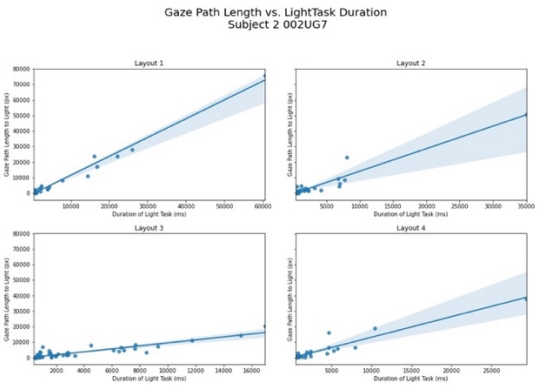
Subject 1



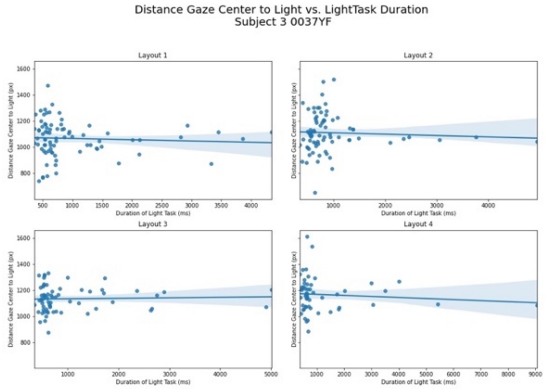


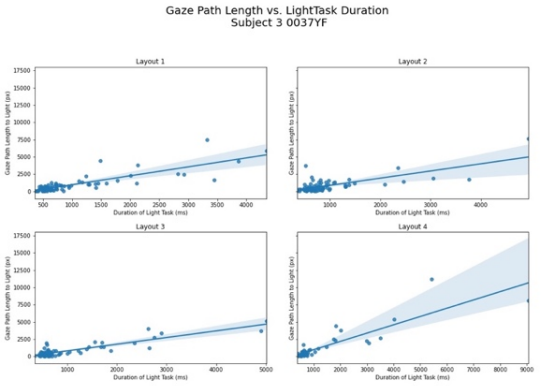
Subject 2



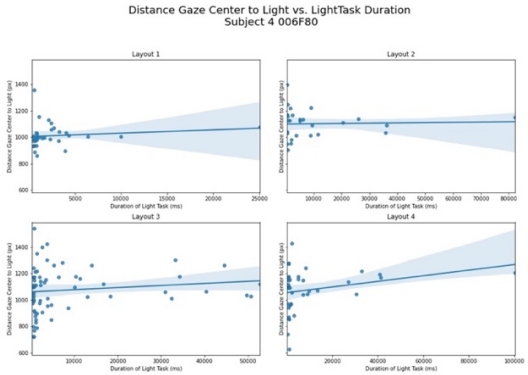


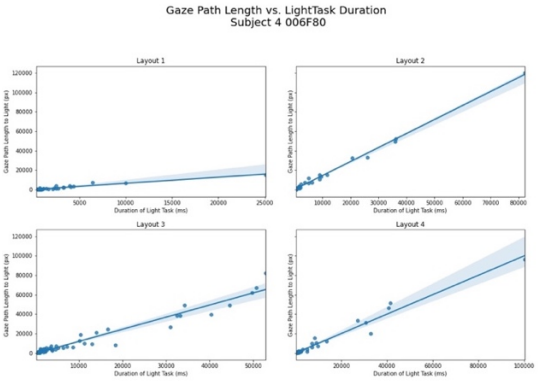
Subject 3



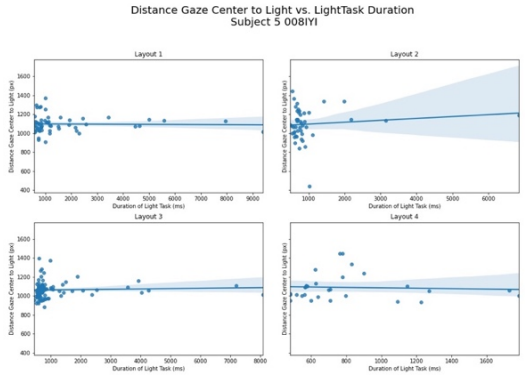


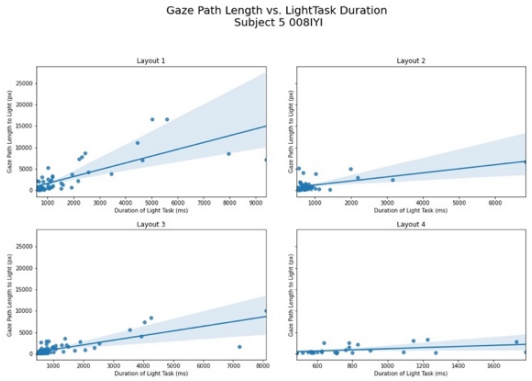
Subject 4



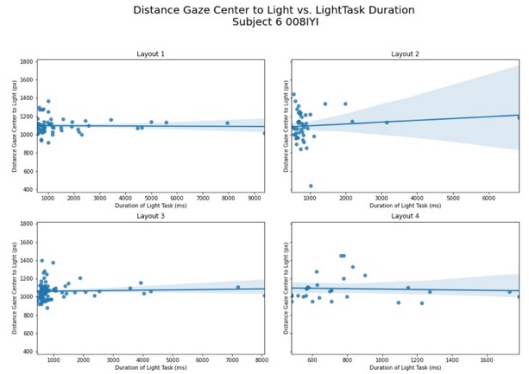


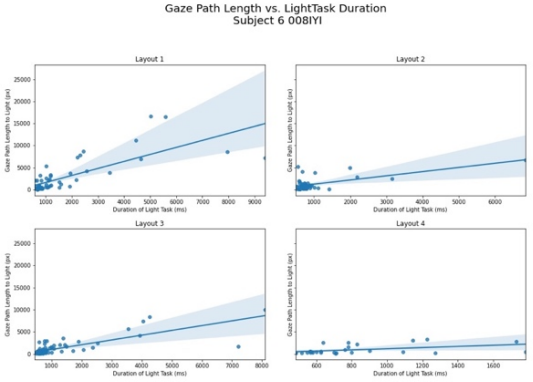
Subject 5



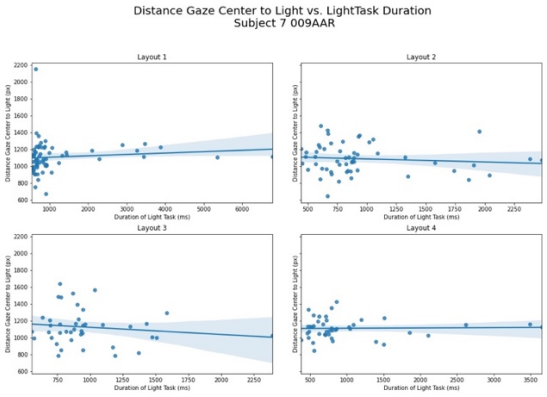


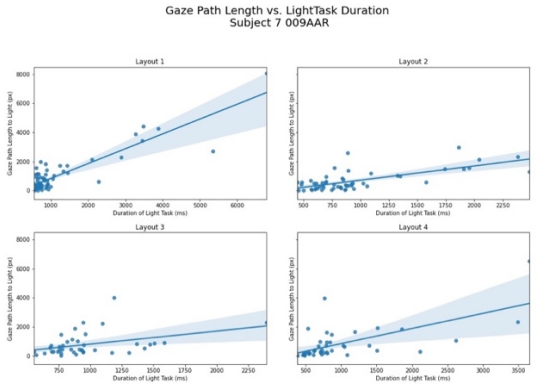
Subject 6



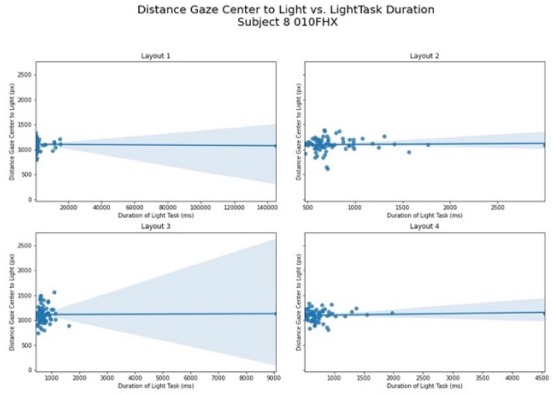


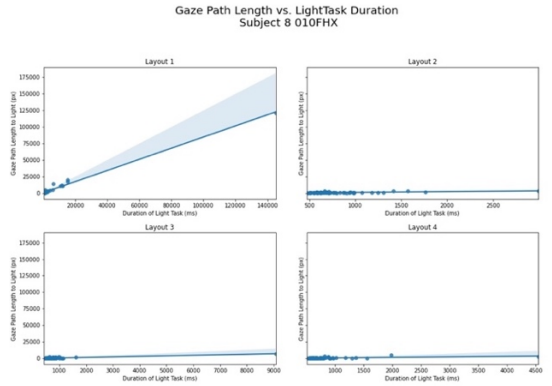
Subject 7



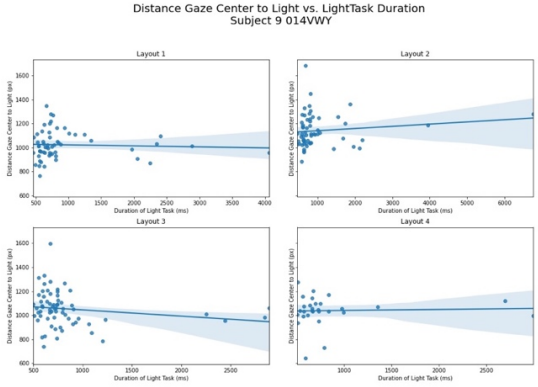


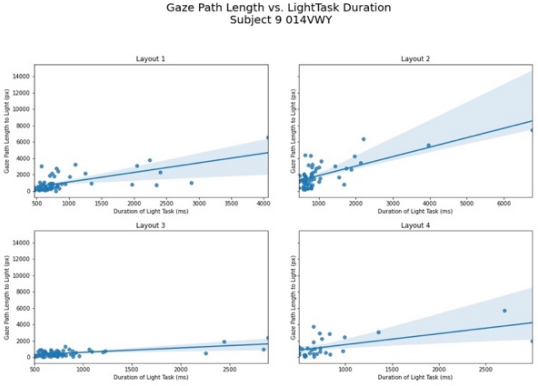
Subject 8



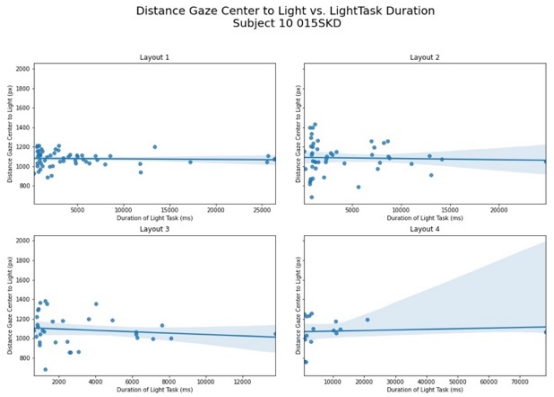


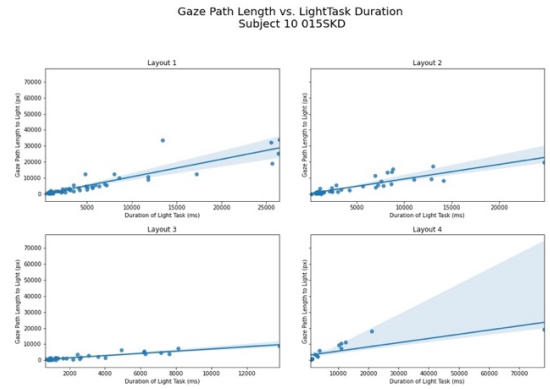
Subject 9



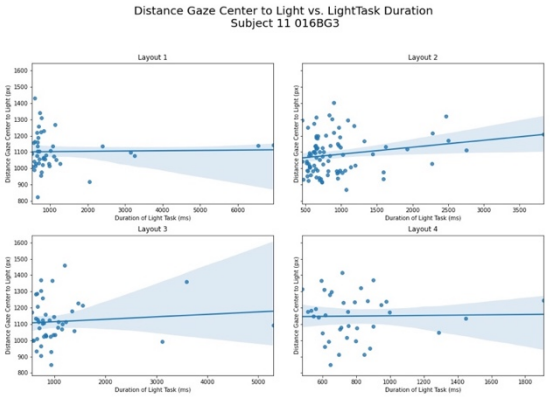


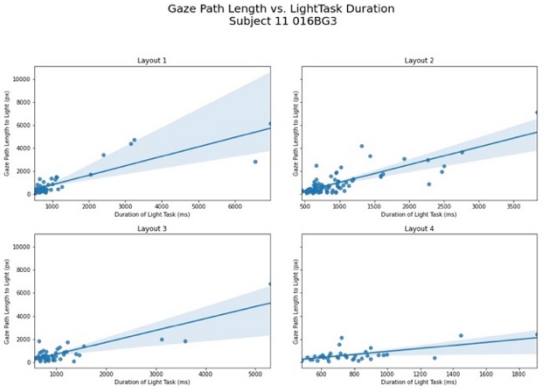
Subject 10



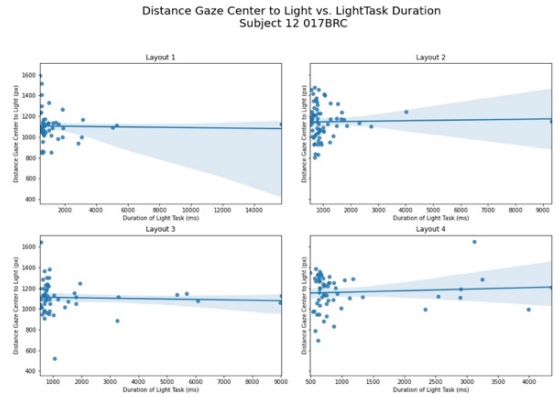


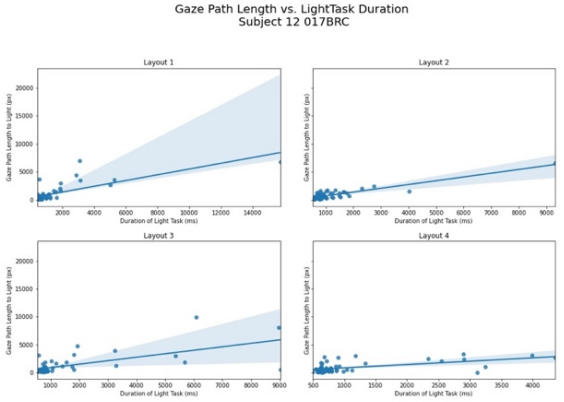
Subject 11



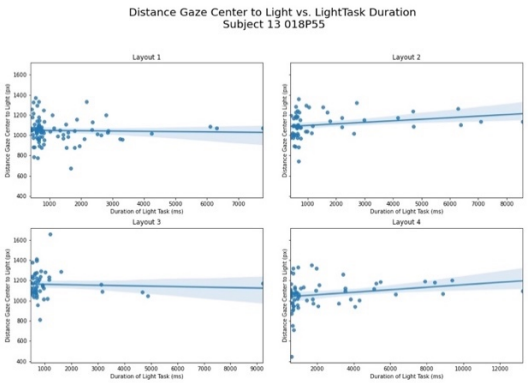


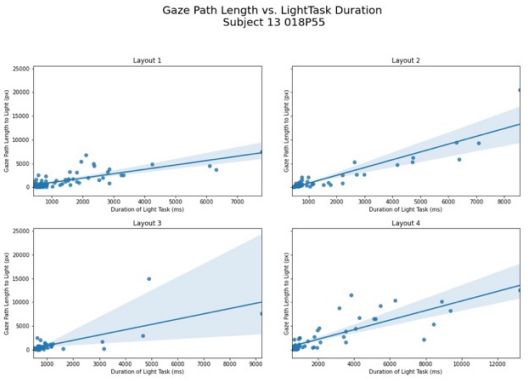
Subject 12



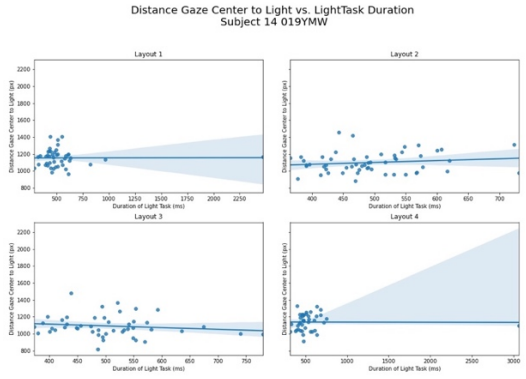


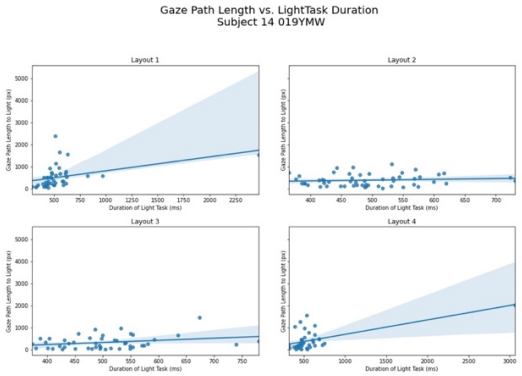
Subject 13



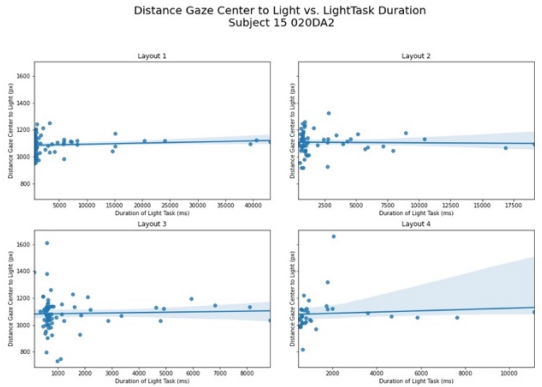


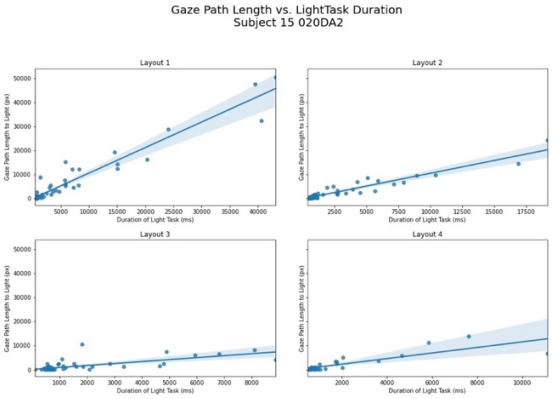
Subject 14



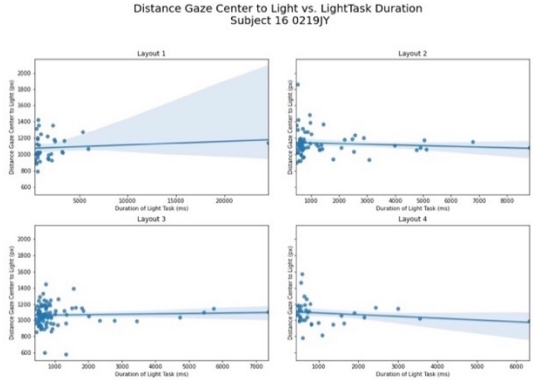


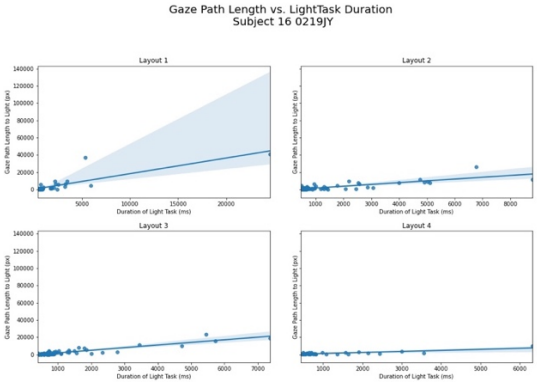
Subject 15



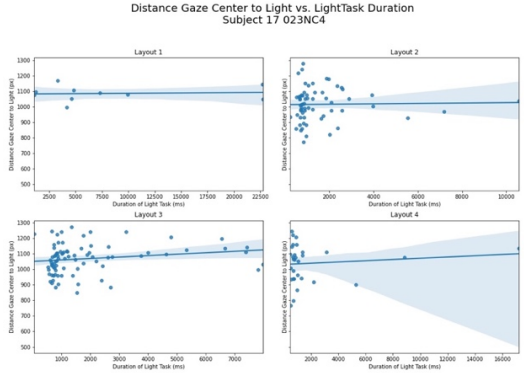


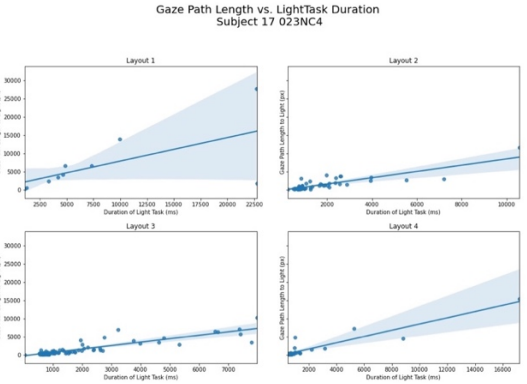
Subject 16



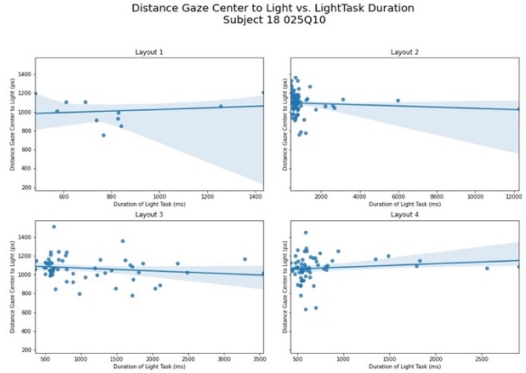


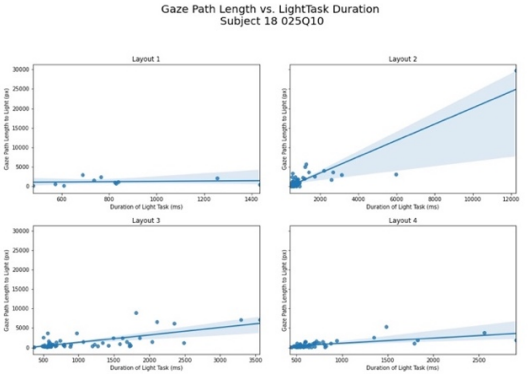
Subject 17



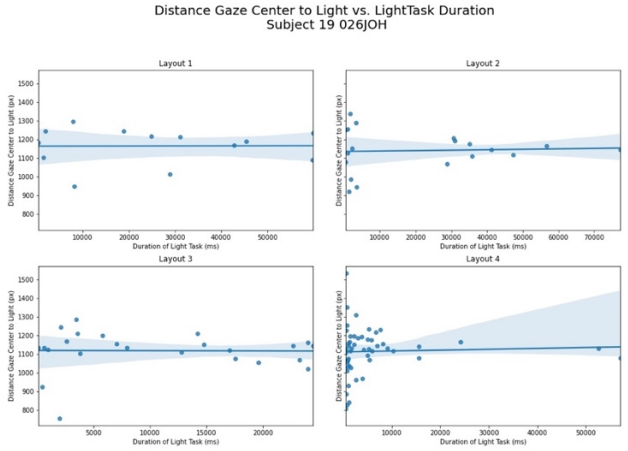


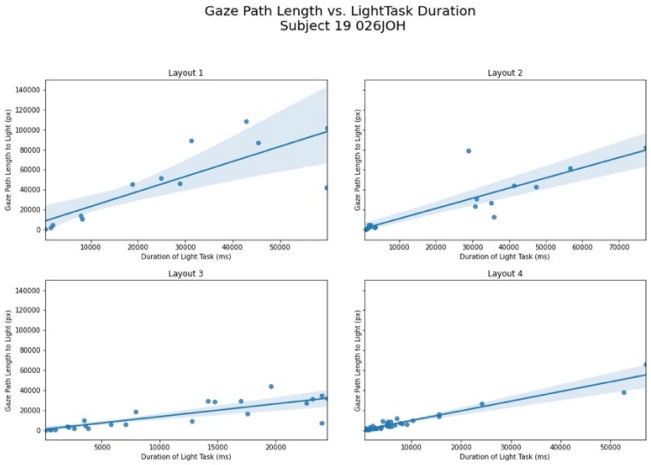
Subject 18



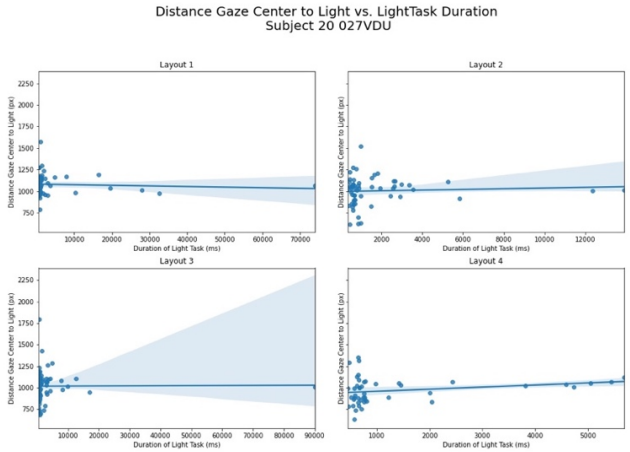


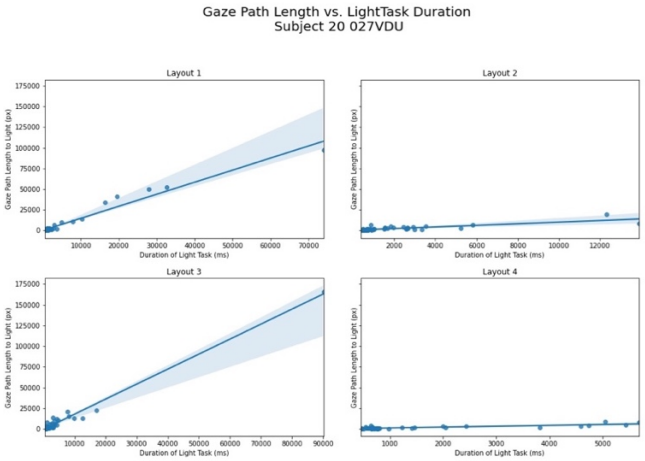
Subject 19





Subject 20





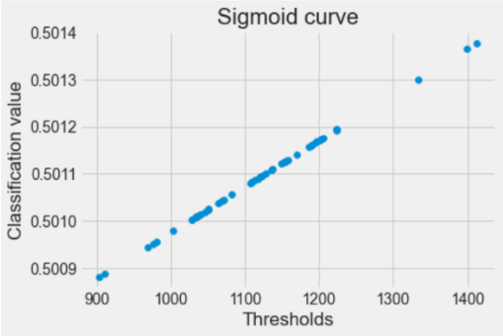
10.8 Appendix 7: Additional Logistic Result Visualizations

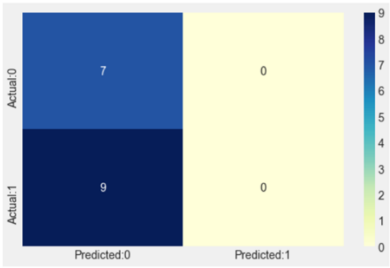
Table 40. Appendix Table: Additional Logistic Regression Results

ML LogisticRegression- Distance to Centroid vs. Duration

Target Variable	Target Outcomes	Independent Features	Dataset
Response Time	Less Than 1000ms More Than 1000ms	Distance from Centroid to Light (dst)	biglights

```
Logistic Model Intercept: [-3.46244469e-09]
Logistic Model Coefficient: [[3.9013933e-06]]
Decision threshold: [[0.00088749]]
```

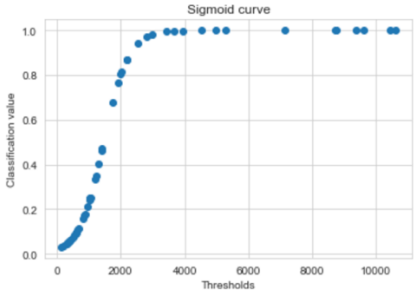


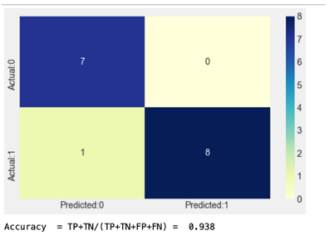



```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.438
```

ML-LogisticRegression Duration

Target Variable	Target Outcomes	Independent Features	Dataset
Response Time	Less Than 1000ms More Thank 1000ms	Total Path Travelled (lng)	biglights

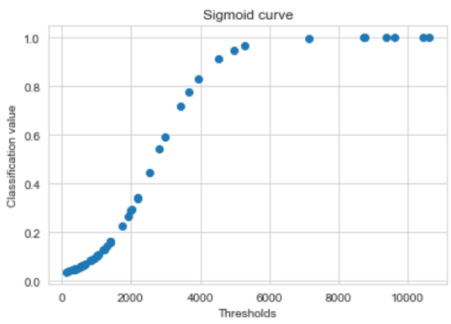


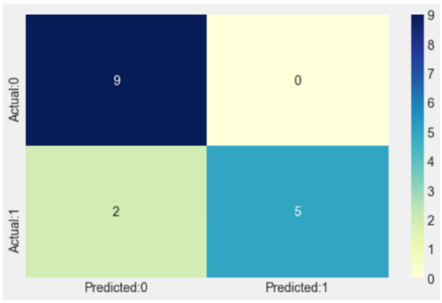



```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.938
```

```
Optimization terminated successfully.
         Current function value: 0.555225
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                   52
Model:                          Logit   Df Residuals:                       51
Method:                           MLE   Df Model:                            0
Date:                Wed, 11 Jan 2023   Pseudo R-squ.:                  0.1990
Time:                        20:38:33   Log-Likelihood:                -28.872
converged:                       True   LL-Null:                       -36.044
Covariance Type:            nonrobust   LLR p-value:                       nan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
lng            0.0004      0.000      2.629      0.009       0.000       0.001
==============================================================================
```

Target Variable	Target Outcomes	Independent Features	Dataset
Response Time	Less Than 2000ms	Total Path Travelled (lng)	biglights
	More Thank 2000ms		





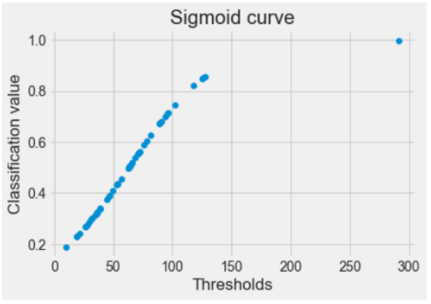

```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.875
```

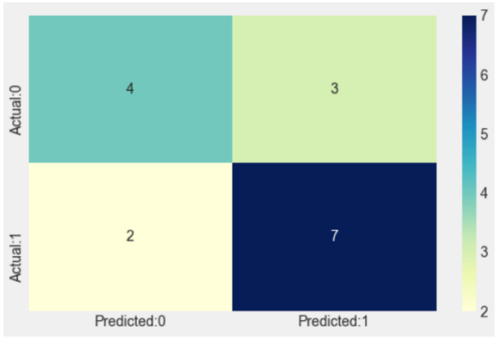
```
Optimization terminated successfully.
         Current function value: 0.643039
         Iterations 4
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                   52
Model:                          Logit   Df Residuals:                       51
Method:                           MLE   Df Model:                            0
Date:                Wed, 11 Jan 2023   Pseudo R-squ.:                0.003091
Time:                        20:44:21   Log-Likelihood:                -33.438
converged:                       True   LL-Null:                       -33.542
Covariance Type:            nonrobust   LLR p-value:                       nan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
lng            0.0002   9.27e-05      2.007      0.045    4.32e-06       0.000
==============================================================================
```

Target Variable	Target Outcomes	Independent Features	Dataset
Response Time	Less Than 1000ms	Gaze Dispersion Y	biglights
	More Than 1000ms		

ML-LogisticRegression Duration vs. Path

```
Logistic Model Intercept: [-1.73222514]
Logistic Model Coefficient: [[0.02751592]]
Decision threshold: [[62.95355064]]
```

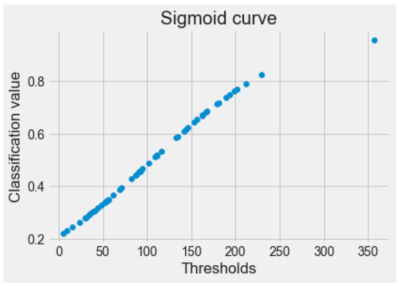


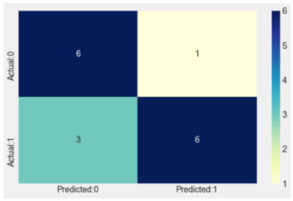



```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.688
```

Target Variable	Target Outcomes	Independent Features	Dataset
Response Time	Less Than 1000ms	Gaze Dispersion X	biglights
	More Thank 1000ms		

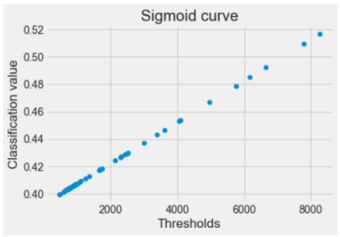
```
Logistic Model Intercept: [-1.32081577]
Logistic Model Coefficient: [[0.01250631]]
Decision threshold: [[105.61195302]]
```



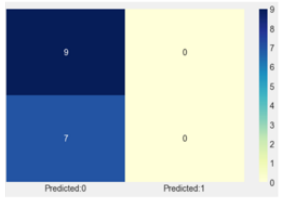



```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.75
```

Target Variable	Target Outcomes	Independent Features	Dataset
Movement Direction	moved away from the light	Speed , Full Path (lng_rate)	biglights
	moved towards the light		

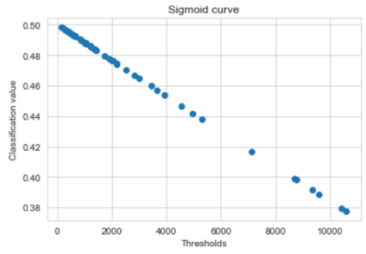



```
Optimization terminated successfully.
         Current function value: 0.680300
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                    pos   No. Observations:                   52
Model:                          Logit   Df Residuals:                       51
Method:                           MLE   Df Model:                            0
Date:                Wed, 11 Jan 2023   Pseudo R-squ.:                0.001417
Time:                        19:59:23   Log-Likelihood:                -35.376
converged:                       True   LL-Null:                       -35.426
Covariance Type:            nonrobust   LLR p-value:                       nan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
lng_rate       5.7693      6.152      0.938      0.348      -6.288      17.827
==============================================================================
```

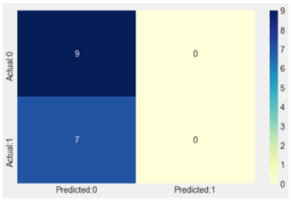



```
uracy  = TP+TN/(TP+TN+FP+FN) =  0.562
```

Target Variable	Target Outcomes	Independent Features	Dataset
Movement Direction	moved away from the light	Total Path Travelled (lng)	biglights
	moved towards the light		

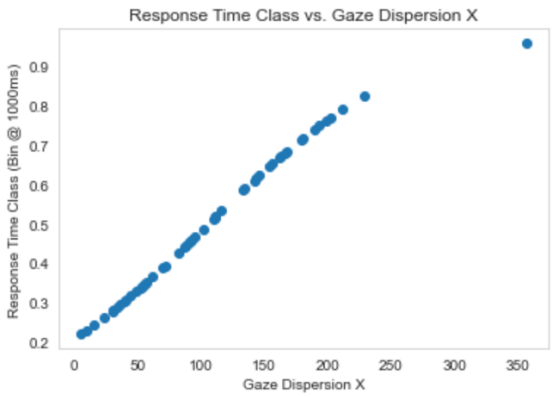



```
Optimization terminated successfully.
         Current function value: 0.689028
         Iterations 3
                           Logit Regression Results
==============================================================================
Dep. Variable:                    pos   No. Observations:                   52
Model:                          Logit   Df Residuals:                       51
Method:                           MLE   Df Model:                            0
Date:                Wed, 11 Jan 2023   Pseudo R-squ.:                -0.01139
Time:                        20:03:57   Log-Likelihood:                -35.829
converged:                       True   LL-Null:                       -35.426
Covariance Type:            nonrobust   LLR p-value:                       nan
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
lng        -4.712e-05   7.27e-05     -0.648      0.517      -0.000    9.54e-05
==============================================================================
```

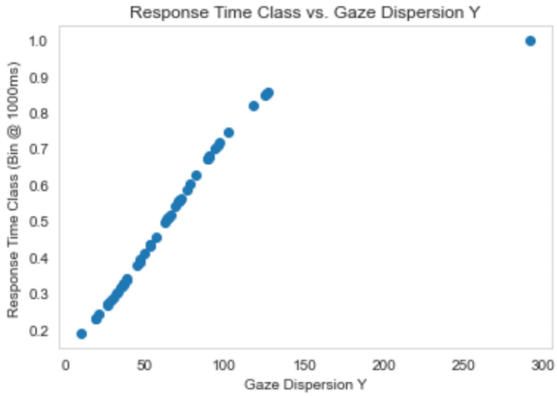



```
Accuracy  = TP+TN/(TP+TN+FP+FN) =  0.562
```

Response Time Logistic Regression: (1s split) vs. Gaze Dispersion (a) -X (b) -Y

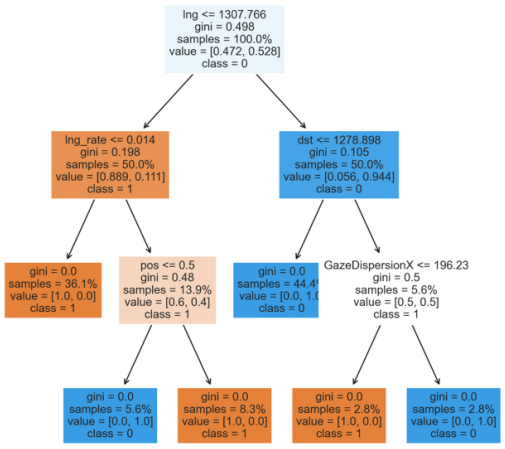


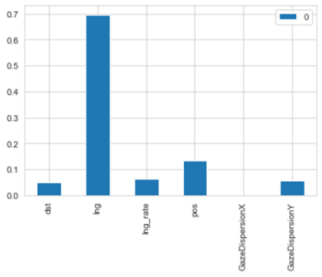
Logistic Model Intercept: [-1.32081577]
Logistic Model Coefficient: [[0.01250631]]



Logistic Model Intercept: [-1.73222514]
Logistic Model Coefficient: [[0.02751592]]

10.9 Appendix 8: Decision Tree Hyperparameter Tuning

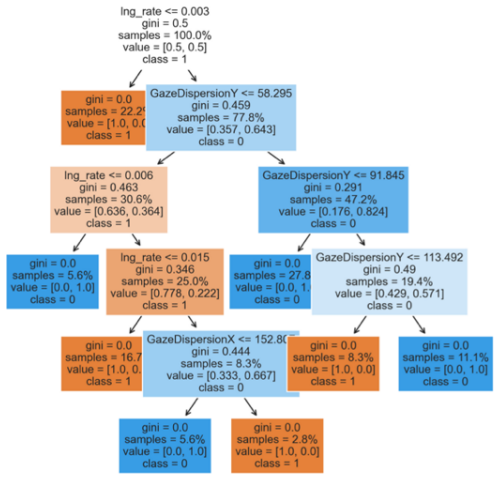


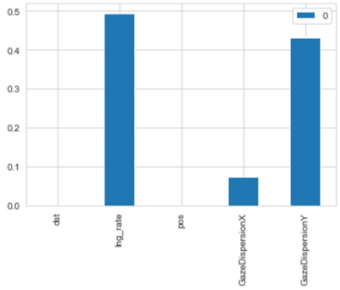



```
accuracy score: 0.875

Precision score: 0.8571428571428571

Recall score: 0.8571428571428571

Confusion Matrix:
 [[8 1]
 [1 6]]
```

ccp_aplpha = 0.01

Removing Total Length to evaluate remaining parameters





	0
dst	0.000000
lng_rate	0.494468
pos	0.000000
GazeDispersionX	0.074074
GazeDispersionY	0.431457

```
accuracy score: 0.6875

Precision score: 0.6666666666666666

Recall score: 0.75

Confusion Matrix:
 [[5 3]
 [2 6]]
```

ccp_aplpha = 0.02

10.10 Appendix 9: Annotated File Structure

Table 41. Appendix Table: Data processing and transformations

2D_002UG7_04-19-2021_13.46_Gaze_BackFill

timeStamp	intervalDuration	headPosX	headPosY	headPosZ	headRotEulerX	headRotEulerY	headRotEulerZ	screenGazePosX	screenGazePosY	screenGazePosZ	worldGazePosX	worldGazePosY
674	51	-44.48016	2.13545	16.18486	13.0595	282.4457	3.282526	1058.852	1343.05	7.309199	-51.73469	1.339393
720	39	-44.6217	2.137506	16.17068	12.68904	283.3853	2.993992	1051.863	1347.953	7.159662	-51.73555	1.443127
778	50	-44.74694	2.138559	16.15658	12.54279	283.822	2.698383	1053.724	1347.658	7.046451	-51.73677	1.474252
827	42	-44.88858	2.140581	16.14246	12.22037	284.3052	2.753803	1113.998	1396.584	0.8706633	-45.74773	2.114331
874	40	-45.01385	2.140422	16.13286	12.22139	284.685	3.032617	1054.386	1383.541	6.731932	-51.73752	1.813938
925	44	-45.13151	2.139576	16.1265	12.49395	284.9868	3.101391	1037.067	1386.64	6.58023	-51.7337	1.804208
982	50	-45.25964	2.138909	16.12222	12.59114	285.1104	3.008605	1041.924	1391.698	6.45703	-51.73447	1.838394
1058	69	-45.40036	2.137469	16.11954	12.77397	285.6493	2.768026	1108.321	1445.852	0.8664609	-46.2625	2.150689
1148	84	-45.53109	2.138277	16.11977	12.4435	286.3608	2.778245	1096.401	1434.435	0.8777472	-46.40219	2.144994
1196	41	-45.75822	2.139643	16.12277	12.14615	287.1521	2.6374	1116.898	1420.456	0.7875837	-46.5287	2.138313
1259	56	-45.92686	2.140894	16.12938	11.84536	288.5822	2.458936	1084.173	1420.826	0.8414274	-46.75329	2.143014
1325	58	-46.08306	2.141559	16.14595	11.88293	293.2189	2.410475	1009.115	1434.516	0.8441845	-46.92091	2.153207
1381	49	-46.2351	2.140434	16.18862	12.5337	307.4452	4.118548	960.4156	1430.645	0.8332583	-47.00817	2.132106
1428	40	-46.33981	2.140903	16.24592	12.23588	314.6454	5.299827	841.5267	1438.067	0.6960315	-47.00817	2.132106
1484	49	-46.43576	2.144264	16.33407	10.82721	329.5574	5.347037	484.7918	1439.006	0.4954236	-47.00817	2.132106
1611	64	-46.54186	2.143779	16.58	10.96757	351.695	3.91097	944.0062	1196.172	4.886823	-48.14958	0.9298215
1686	68	-46.53595	2.142574	16.58214	11.00973	355.2365	4.365383	1160.327	1126.406	4.723908	-46.60391	0.6737746
1732	39	-46.52689	2.140275	16.58524	11.1381	0.1285429	4.043436	1224.763	1118.319	1.961348	-46.24854	1.517674
1780	40	-46.52528	2.137714	16.58574	12.05421	359.4846	3.988281	1179.683	1120.778	3.103722	-46.28054	1.102331
1831	43	-46.52724	2.136964	16.58619	12.40457	357.7538	3.641337	1121.148	1191.643	4.785124	-46.64227	0.8727852
1901	61	-46.50943	2.137668	16.59862	12.35526	352.303	2.834645	1087.266	1161.711	4.773818	-47.23719	0.7091856
1945	38	-46.55398	2.137482	16.62662	12.38943	349.2002	2.204545	1122.04	1190.198	4.79018	-47.35632	0.8640115
1995	38	-46.70976	2.138536	16.70041	12.56196	339.837	1.754641	1317.272	1170.73	4.546767	-47.19292	0.8468571
2042	40	-46.83406	2.139453	16.73534	12.59501	332.2069	1.392683	1351.938	1168.102	4.498184	-47.73597	0.8433764
2095	46	-47.00392	2.140656	16.74615	12.1123	312.2586	359.8827	1745.945	1132.593	3.748921	-47.79733	0.9098822
2157	55	-47.15409	2.144337	16.69573	10.0195	290.0088	358.5659	2145.501	993.1228	2.947025	-48.53265	0.7293015
2218	54	-47.30102	2.145269	16.58522	9.658648	280.8239	358.7834	1926.642	1060.86	1.927238	-48.75686	1.398903
3587	48	-50.15122	2.135094	15.64036	10.29566	258.1862	2.211751	1299.228	1195.525	6.583005	-56.741	0.7095345
3649	55	-50.29032	2.135384	15.60853	10.09093	253.6293	1.903294	1096.244	1207.465	0.8678573	-51.10102	1.95461
3774	64	-50.56882	2.133819	15.53343	10.98279	253.4699	3.374373	1204.917	1152.244	1.103553	-51.62465	1.825611
3949	53	-51.00455	2.132058	15.46647	11.75924	257.0469	4.013899	1213.853	1164.194	6.606341	-57.39245	0.3029193
4019	62	-51.07924	2.133842	15.49867	11.58794	257.7839	3.616225	1206.449	1158.459	6.496129	-57.35117	0.3029193
4077	52	-51.2117	2.136901	15.59104	12.09992	260.0601	2.839954	1152.407	1184.326	6.586662	-57.53376	0.3864887
4138	54	-51.32542	2.214626	15.66556	12.00986	262.15	2.815243	1136.133	1162.95	6.607463	-57.64348	0.3029194
4522	74	-52.11474	2.127198	15.8295	13.26658	256.2612	1.138611	897.9464	1108.267	0.8924639	-52.87629	1.789739
4570	41	-52.29515	2.12737	15.85099	13.70928	261.7683	2.413432	775.7787	1093.45	0.8083146	-52.9938	1.793507
4624	47	-52.40331	2.12824	15.87877	13.6366	265.9268	3.248189	790.6597	1129.653	0.8719932	-53.1965	1.800677
4777	47	-52.70989	2.148379	16.04045	11.84152	290.5586	3.417005	957.2486	1122.899	4.246522	-56.74099	0.6895896
5183	42	-52.66925	2.160312	16.85603	7.376723	22.05879	357.4933	1281.959	1277.962	0.7367527	-52.25781	2.09488
5226	36	-52.58587	2.158502	16.92274	7.661511	28.3411	356.9704	1334.946	1283.12	0.4714968	-52.2547	2.114715
5269	36	-52.50119	2.152938	16.96845	9.386404	38.18645	357.3078	1266.206	1300.896	0.3263287	-52.25332	2.12117
5325	48	-52.41025	2.152271	17.00052	10.07822	46.00908	357.6379	935.6727	1393.834	0.2623898	-52.25386	2.155673
5373	42	-52.29636	2.154137	17.03264	9.683545	49.5381	356.8918	650.5381	1331.777	0.1027561	-52.25261	2.150996

timeStamp	worldGazePosZ	leftPupilDiameter	rightPupilDiameter	avgPupilDiameter	leftEyeOpenness	rightEyeOpenness	avgOpenness	light	dx	dy	dt	d_travel	speed	LightStatus
674	17.27322	5.331543	5.228943	5.280243	1.0	1.0	1.0	-1.0						OFF
720	17.30223	5.1819	5.32402	5.25296	1.0	1.0	1.0	-1.0	-6.98900	4.903000	46.0	8.537302267	0.18559	OFF
778	17.34331	5.165131	5.372299	5.268715	1.0	1.0	1.0	-1.0	1.860999	-0.29500	58.0	1.884236184	0.03248	OFF
827	16.35559	5.129822	5.324432	5.227127	1.0	1.0	1.0	-1.0	60.27400	48.92600	49.0	77.63187845	1.58432	OFF
874	17.36869	5.143402	5.342224	5.242813	1.0	1.0	1.0	-1.0	-59.6120	-13.0430	47.0	61.02221229	1.29834	OFF
925	17.23983	5.134964	5.242203	5.188583	1.0	1.0	1.0	-1.0	-17.3190	3.099000	51.0	17.59407746	0.34498	OFF
982	17.26563	5.178833	5.083939	5.131386	1.0	1.0	1.0	-1.0	4.856999	5.057999	57.0	7.012404223	0.12302	OFF
1058	16.34667	5.215836	5.108673	5.162254	1.0	1.0	1.0	-1.0	66.39699	54.15400	76.0	85.68090408	1.12738	OFF
1148	16.34883	5.210373	5.171173	5.190773	1.0	1.0	1.0	-1.0	-11.9199	-11.4170	90.0	16.50558357	0.18339	OFF
1196	16.35714	5.153336	5.0746	5.113968	1.0	1.0	1.0	-1.0	20.49699	-13.9790	48.0	24.81006751	0.51687	OFF
1259	16.37051	5.18959	5.163879	5.176735	1.0	1.0	1.0	-1.0	-32.7249	0.370000	63.0	32.72709160	0.51947	OFF
1325	16.38724	5.253647	5.23053	5.242088	0.7871476	0.9336692	0.8604084	-1.0	-75.0580	13.69000	66.0	76.29626114	1.15600	OFF
1381	16.58003	-1.0	-1.0	-1.0	0.446156	0.4832151	0.4646856	-1.0	-48.6994	-3.87100	56.0	48.85300606	0.87237	OFF
1428	16.58003	-1.0	-1.0	-1.0	0.446156	0.4832151	0.4646856	-1.0	-118.888	7.422000	47.0	119.1203451	2.53447	OFF
1484	16.58003	5.240463	5.421494	5.330978	1.0	1.0	1.0	-1.0	-356.734	0.939000	56.0	356.7361358	6.37028	OFF
1611	21.13806	5.494003	5.387634	5.440819	1.0	1.0	1.0	-1.0	459.2144	-242.834	127.0	519.4672431	4.09029	OFF
1686	21.11888	5.494003	5.387634	5.440819	1.0	1.0	1.0	-1.0	216.3208	-69.7660	75.0	227.2927259	3.03056	OFF
1732	18.46104	5.205765	5.145645	5.175705	1.0	1.0	1.0	-1.0	64.43599	-8.08699	46.0	64.94149416	1.41177	OFF
1780	19.54066	5.33374	5.182083	5.257912	1.0	1.0	1.0	-1.0	-45.0799	2.459000	48.0	45.14701630	0.94056	OFF
1831	21.20668	5.247162	5.141052	5.194107	1.0	1.0	1.0	-1.0	-58.5350	70.86500	51.0	91.91406013	1.80223	OFF
1901	21.11594	5.293442	5.20845	5.250946	1.0	1.0	1.0	-1.0	-33.8819	-29.9320	70.0	45.20967316	0.64585	OFF
1945	21.18161	5.214157	5.316986	5.265572	1.0	1.0	1.0	-1.0	34.77399	28.48700	44.0	44.95264447	1.02165	OFF
1995	21.17878	5.275589	5.397537	5.336563	1.0	1.0	1.0	-1.0	195.2320	-19.4680	50.0	196.2002468	3.92400	OFF
2042	21.14276	5.204285	5.563263	5.383774	1.0	1.0	1.0	-1.0	34.66600	-2.62799	47.0	34.76547051	0.73969	OFF
2095	21.18188	5.402115	5.387939	5.395027	1.0	1.0	1.0	-1.0	394.0070	-35.5090	53.0	395.6038487	7.46422	OFF
2157	20.9256	5.112106	5.25238	5.182243	0.7679388	0.7005725	0.7342557	-1.0	399.5560	-139.470	62.0	423.1984567	6.82578	OFF
2218	18.7045	5.112106	5.25238	5.182243	0.7679388	0.7005725	0.7342557	-1.0	-218.859	67.73719	61.0	229.1017026	3.75576	OFF
3587	15.73031	5.671402	5.863464	5.767433	0.8408081	0.9565305	0.8986692	0.0	-627.414	134.6650	1369.0	641.7031943	0.46873	ON
3649	15.35488	5.671936	5.731293	5.701614	0.9175768	0.9869552	0.952266	0.0	-202.984	11.93999	62.0	203.3348663	3.27959	ON
3774	15.35021	5.705063	5.771347	5.738205	0.9960725	0.9267777	0.9614251	0.0	108.673	-55.2210	125.0	121.8982353	0.97518	ON
3949	14.8337	5.716415	5.800583	5.758499	0.9923062	0.9469532	0.9696297	0.0	8.936000	11.95000	175.0	14.92161506	0.08526	ON
4019	14.90345	5.76445	5.841888	5.803169	0.9246712	0.9497868	0.937229	0.0	-7.40400	-5.73499	70.0	9.365331868	0.13379	ON
4077	14.81524	5.798492	5.795456	5.796974	0.8905839	0.9756615	0.9331226	-1.0	-54.0420	25.86700	58.0	59.91359990	1.03299	OFF
4138	15.00821	5.832916	5.956131	5.894524	0.8870326	0.9831729	0.9351027	-1.0	-16.2739	-21.3760	61.0	26.86589756	0.44042	OFF
4522	15.41847	5.950897	5.73465	5.842773	1.0	1.0	1.0	-1.0	-238.186	-54.6830	384.0	244.3830741	0.63641	OFF
4570	15.43809	5.837875	5.58374	5.710808	1.0	1.0	1.0	-1.0	-122.167	-14.8170	48.0	123.0629530	2.56381	OFF
4624	15.50381	5.84935	5.63652	5.742935	1.0	1.0	1.0	-1.0	14.88100	36.20300	54.0	39.14206650	0.72485	OFF
4777	16.77691	-1.0	5.143509	2.071754	0.2886642	0.42082	0.3547421	-1.0	166.5889	-6.75400	153.0	166.7257572	1.08971	OFF
5183	17.48175	-1.0	5.78009	2.390045	0.6484237	0.4510852	0.5497544	-1.0	324.7104	155.0630	406.0	359.8352092	0.88629	OFF
5226	17.27797	-1.0	5.963196	2.481598	0.6484237	0.4510852	0.5497544	-1.0	52.98699	5.157999	43.0	53.23745986	1.23808	OFF
5269	17.18762	-1.0	5.963196	2.481598	0.6484237	0.4510852	0.5497544	-1.0	-68.7400	17.77600	43.0	71.00122376	1.65119	OFF
5325	17.22311	-1.0	5.963196	2.481598	0.6484237	0.4510852	0.5497544	-1.0	-330.533	92.93800	56.0	343.3507452	6.13126	OFF
5373	17.14115	-1.0	5.800049	2.400024	1.0	1.0	1.0	-1.0	-285.134	-62.0570	48.0	291.8095463	6.07936	OFF

2D_002UG7_04-19-2021_13.30_LightFilter

timeStamp	timeDuration	lightCorrectness	dst	dst_s	dst_e	dst_d	lng	lng_rate
25957	25944	TRUE	1146.4148542983700	1207.0462244379900	1272.9477273413200	65.901502903336	28026.645394377400	1.0797336130669000
59363	16765	TRUE	1171.990728239080	1187.3413473306700	1226.6461102816100	39.304762950943100	16564.66805505450	0.279040278541423
71378	399	TRUE	1240.3465429141900	1243.820976910170	1237.8922197975300	-5.92875711263855	92.58514051726660	0.00129711032134925O
80425	800	TRUE	1157.2830808450200	1144.5738374875600	1159.345478872020	14.771641384462800	246.49143060142200	0.0030648608094674800
86339	3875	TRUE	1009.9399023540300	1006.6204004613700	875.5392416311220	-131.08115883024400	2646.2369845494300	0.030649381907937700
91952	434	TRUE	1047.2149847670000	1112.512381344140	1027.442731167530	-85.06965017660290	107.09458005683400	0.0011646791810600500
96246	1066	TRUE	1068.008726655420	1083.8177527854900	1152.4244959371500	68.60674315165760	437.63703799826400	0.004547067285895140
100407	727	TRUE	1067.1438651839100	1145.5071837138300	1048.195900157030	-97.31128355680720	239.54459769843700	0.0023857360313368300
164246	60494	TRUE	1120.89618192919	1036.0922955110000	1161.0183807623400	124.92608525133900	62138.71455136010	0.378327110257541
204082	7801	TRUE	1064.1403846729400	1085.9850974230700	1062.9490402277100	-23.036057195367000	4526.503897412080	0.02217982917362670
211490	680	TRUE	1087.5747318836400	1057.559009119110	1157.1428674541400	99.58385833503170	180.2630846586260	0.0008523480290256090
216401	806	TRUE	1207.7095800887500	1216.3234445019100	1188.974621600480	-27.348822901435600	573.1398656429210	0.0026485083971096300
219163	674	TRUE	1143.037790534350	1289.2109581787200	1127.7753998208200	-161.43555835790300	277.3390032312130O	0.0012654462807646000
226747	340	TRUE	1206.1306744152500	1293.1998909625300	1131.8880689719300	-161.31182199059900	206.4242638458790	0.0009103726349009210
231493	646	TRUE	1084.8339177157800	1193.5934070436200	1181.2718641642200	-12.321542879406300	1001.2255160471700	0.004325079013392080
236260	677	TRUE	1091.640625153230	1054.995153001660	1242.4562631909500	187.46111018928400	626.150338528916	0.0026502596229955000
255296	16012	TRUE	1100.3938826812600	1124.126174047200	1286.335814317560	162.20964027035400	23647.889655928700	0.09262929954221270
288630	22041	TRUE	1168.393006068180	986.6344568263370	968.3953943219680	-18.23906250436840	20615.631778049800	0.07142581082371830
312079	580	TRUE	1172.2185548031200	1205.2949075840300	1157.966124480760	-47.328783103267800	323.4175643206670	0.0010363323527717900
316765	463	TRUE	895.3606956651570	957.5316596985190	955.7181922188150	-1.8134674797037200	365.19249475469400	0.001152881457088680
320164	2299	TRUE	1149.5473165332900	1087.6185229353200	1232.5263533843900	144.90783044906900	645.9308773206460	0.0020175000228653000
326002	664	TRUE	1021.6983634570000	909.8206176923010	1154.224773553660	244.4041558613600	354.3657969296400	0.0010870049782812400
330535	471	TRUE	918.2910267392460	910.1363842348020	914.0407110025240	3.9043267677217300	85.23242349357760	0.00025786202215673900
341666	833	TRUE	1023.445985947950	967.1903278972550	1145.2926429280000	178.1023150307430	308.2540496765920	0.0009022087350704840
343741	1075	TRUE	1155.4061875145600	1180.6169511505400	1070.9639093125400	-109.65304183800100	643.5973244533480	0.001872332146742310
354585	329	TRUE	1120.4253634035800	1142.501352426330	1106.300095103040	-36.201257323291000	42.79612120186840	0.00012069354654559100
358304	607	TRUE	1075.828152716150	1050.7805422660800	930.2038462751050	-120.57669599097600	355.23915443918200	0.0009914462424063990
370558	2228	TRUE	1028.855701914200	994.1600817182310	888.1192963470620	-106.04078537116900	1760.7252285608700	0.004751550981387180O

S01_001RL5-L1_LightTask_Out_noNA

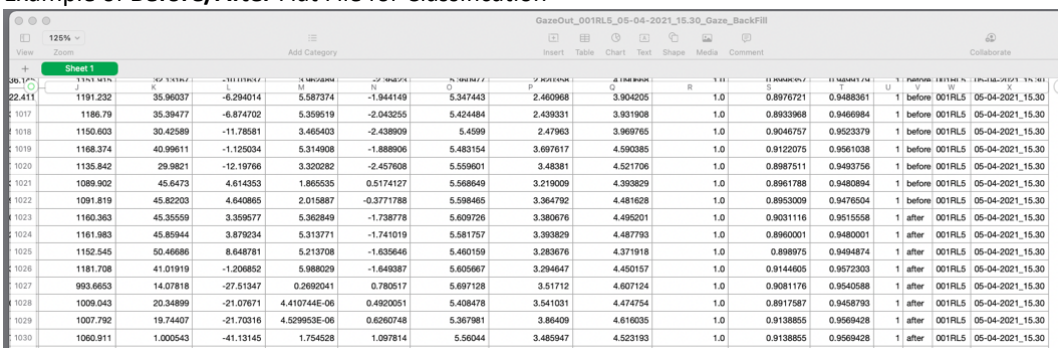
	timeStamp	timeDuration	lightCorrectness	dst	lng	Dist / Dur	Length / Dur
0	2140	1096	TRUE	1071.515852	1891.07125	0.977660449	1.725429973
1	15259	7995	TRUE	1163.232081	16277.00579	0.145494944	2.035898161
2	24269	810	TRUE	1280.134794	2071.803723	1.580413325	2.557782374
3	41504	5900	TRUE	1154.131863	7489.60332	0.19561557	1.269424292
4	48112	3487	TRUE	1210.638161	5339.580246	0.347186166	1.531281975
5	58092	2770	TRUE	1129.523387	4749.680864	0.407770176	1.714686232
6	71805	12699	TRUE	1203.989845	20006.64939	0.094809815	1.575450775
7	78628	1700	TRUE	1185.631481	3415.722154	0.697430283	2.009248326
8	86797	2088	TRUE	1199.7163	2616.571167	0.574576772	1.25314711
9	113531	15587	TRUE	1167.840952	29638.48365	0.074924036	1.901487371
10	124212	1552	TRUE	1190.11531	2514.716287	0.766826875	1.620306886
11	130141	3919	TRUE	1162.534953	5577.872975	0.296640713	1.423289864

AggregateML_2D_LightFilter_Disp

	timeStamp	timeDuration	lightCorrectness	dst	dst_s	dst_e	dst_d	lng	lng_rate	GazeDispersionX	GazeDispersionY
0	5169.0	2049.0	TRUE	907.0660696792740	743.5468046417790	930.6624430141150	187.115638372336000	3088.001346393410	0.5974078828387340	150.9426676343160	125.62865468954
1	8456.0	2244.0	TRUE	872.3064608997940	899.0736873521550	1199.0431561028200	299.969468750660000	3378.376246968620	0.3995241540880580	146.5598754628410	53.49241431981060
2	11833.0	1346.0	TRUE	1059.6511578629100	1099.904716064080	1093.8968398825400	-6.0078761815414100	903.1213038165430	0.0763222601044995	50.31299459154890	171.50293046401200
3	19879.0	767.0	TRUE	1271.3915449483500	1137.4190430624100	977.4617718934080	-159.957271168999000	1827.573086009980	0.0919348602047374	159.562861268860800	57.93397252491270
4	22820.0	785.0	TRUE	1021.7275284825200	935.9522107805500	989.835560594222	53.883449813671600	362.568732855532	0.0158882003880601	44.69902450041710	21.48185723633360
5	24976.0	1099.0	TRUE	1108.5390876320700	1029.1164333665100	1151.6526024986900	122.536169132181000	3198.475175077720	0.1280619464717220	250.6238540967150	41.06222275379840
6	40499.0	809.0	TRUE	928.6779523891590	889.5209865433190	1047.416749663670	157.895763120347 0	604.1902777607750	0.0149186426824879	88.446955385700000	43.97991532064510
7	45401.0	830.0	TRUE	1035.679991409430	999.8673635767900	1136.2629303484300	136.395566771640000	1931.316681824400	0.0425391220860430	137.171705719511000	268.86864594562500
8	52502.0	722.0	TRUE	945.5520881146330	978.6142015012860	877.5480825487570	-101.066118952529000	255.111486276132000	0.0048590812974007	62.89935457737220	21.67567642991670
9	59999.0	2344.0	TRUE	1030.578287812780	1093.9422914669700	1049.355086755670	-44.587204711297500	695.405784610339 0	0.0115902895816653	48.552288312768100	44.71658853913690 0
10	65184.0	1969.0	TRUE	985.7716100834570	1101.6377060685600	1017.9297951435600	-83.707910925005300	761.759174886098 0	0.0116862907291068	54.8064905749274	55.7693443824307
11	73408.0	2883.0	TRUE	1010.9320814492200	766.4056085169000	1155.987255574210	389.581647057313 0	999.416946078174	0.0136145508129655	107.4773392213290	40.90149673502550 0
12	78478.0	940.0	TRUE	1163.469955327940	1119.833909387010	1185.7528859387200	65.918976551706000	845.704393302314 0	0.0107763244896953	54.35841371775170	63.40379995630060
13	80261.0	734.0	TRUE	1218.481432018660	1177.1217711813000	1245.327178069380	68.2054068880825	388.220397155366 0	0.0048369743356719	39.71708985256500	64.48583065364760
14	81958.0	628.0	TRUE	1191.820308590800	1058.6547148948000	1131.8568088963400	73.202094001532 70	564.164196590194	0.0068835769124453	71.233131385475960	32.83067173540270
15	86678.0	620.0	TRUE	1031.370316754670	1031.4260272147500	1032.8003813138300	1.37435409098494 00	135.759219470269 00	0.0015662477153403	8.789984676386620	23.05660706163080
16	91530.0	707.0	TRUE	1126.1936966603700	1035.1287295877700	1285.487475443560	250.358745855783 00	424.479252315698 0	0.0046375969880443	123.4327477189260	45.19322778143570
17	93305.0	734.0	TRUE	1276.749412396320	1644.1751414888000	1165.4014461944000	-478.773695294395 0	2176.861180577370	0.0233305951511427	173.849478956593 00	172.65934676955900

Backfill and Overlaps Cleaning

Example of **Before/After** Flat File for Classification



22.411	1191.232	35.96037	-6.294014	5.587374	-1.944149	5.347443	2.460968	3.904205	1.0	0.8976721	0.9488361	1	before	001RL5	05-04-2021_15.30
1017	1186.79	35.39477	-6.874702	5.359519	-2.043255	5.424484	2.439331	3.931908	1.0	0.8933968	0.9466984	1	before	001RL5	05-04-2021_15.30
1018	1150.603	30.42589	-11.78581	3.465403	-2.438909	5.4599	2.47963	3.969765	1.0	0.9046757	0.9523379	1	before	001RL5	05-04-2021_15.30
1019	1168.374	40.99611	-1.125034	5.314908	-1.888906	5.483154	3.697617	4.590385	1.0	0.9122075	0.9561038	1	before	001RL5	05-04-2021_15.30
1020	1135.842	29.9821	-12.19766	3.320282	-2.457608	5.559601	3.48381	4.521706	1.0	0.8987511	0.9493756	1	before	001RL5	05-04-2021_15.30
1021	1089.902	45.6473	4.614353	1.865535	0.5174127	5.568649	3.219009	4.393829	1.0	0.8961788	0.9480894	1	before	001RL5	05-04-2021_15.30
1022	1091.819	45.82203	4.640865	2.015887	-0.3771788	5.598465	3.364792	4.481628	1.0	0.8953009	0.9476504	1	before	001RL5	05-04-2021_15.30
1023	1160.363	45.35559	3.359577	5.362849	-1.738778	5.609726	3.380676	4.495201	1.0	0.9031116	0.9515558	1	after	001RL5	05-04-2021_15.30
1024	1161.983	45.85944	3.879234	5.313771	-1.741019	5.581757	3.393829	4.487793	1.0	0.8960001	0.9480001	1	after	001RL5	05-04-2021_15.30
1025	1152.545	50.46686	8.648781	5.213708	-1.635646	5.460159	3.283676	4.371918	1.0	0.898975	0.9494874	1	after	001RL5	05-04-2021_15.30
1026	1181.708	41.01919	-1.206852	5.988029	-1.649387	5.605667	3.294647	4.450157	1.0	0.9144605	0.9572303	1	after	001RL5	05-04-2021_15.30
1027	993.6653	14.07818	-27.51347	0.2692041	0.780517	5.697128	3.51712	4.607124	1.0	0.9081176	0.9540588	1	after	001RL5	05-04-2021_15.30
1028	1009.043	20.34899	-21.07671	4.410744E-06	0.4920051	5.408478	3.541031	4.474754	1.0	0.8917587	0.9458793	1	after	001RL5	05-04-2021_15.30
1029	1007.792	19.74407	-21.70316	4.529953E-06	0.6260748	5.367981	3.86409	4.616035	1.0	0.9138855	0.9569428	1	after	001RL5	05-04-2021_15.30
1030	1060.911	1.000543	-41.13145	1.754528	1.097814	5.56044	3.485947	4.523193	1.0	0.9138855	0.9569428	1	after	001RL5	05-04-2021_15.30

Event-Level 3D Data frame setup for **T-Test**:

```
nav
```

	timeStamp	TotPathLengthBefore	TotPathLengthAfter	Ang_GzHd_Bef	Ang_GzHd_Aft	AngSpd_GzHd_Bef	AngSpd_GzHd_Aft	RotChange_B	RotChange_A
0	49453	3392.732374	2189.407227	47.953693	10.387180	-3.928302	-0.171726	2.494460	0.612961
1	50467	3562.817195	2009.978875	46.968819	10.286473	-2.097516	-0.010044	2.128513	0.606561
2	261391	799.357963	443.005200	67.652047	53.997810	1.513040	-6.349440	1.462184	2.872409
3	436895	826.244826	514.866227	57.606189	51.518289	11.614662	-2.934520	2.081366	1.893886
4	437611	785.527442	507.179516	57.694256	52.716710	11.381008	-3.256619	2.103316	1.935076

Event-Level 3D Data frame setup for **Machine Learning**:

:

	timeStamp	timeDuration	extraInfo	When	TotPathLength	Ang_GzHd	AngSpd_GzHd	RotChange	HdRotEulerY	avPupDiam	avgOpen
0	146360	307	TURNON	before	264.780233	53.023650	-4.079864	5.605651	280.654199	5.564546	0.897851
1	146360	307	TURNON	after	509.574570	34.117125	-0.430178	1.633355	101.360906	5.497365	0.937564
2	152113	5381	TURNON	before	403.584480	62.327312	-3.389348	3.043912	243.964064	5.636117	0.913321
3	152113	5381	TURNON	after	544.859029	34.023371	-0.447774	1.571252	99.125495	5.353143	0.925635
4	229010	6369	TURNON	before	360.871623	51.258009	1.558930	1.735274	254.732936	5.178769	0.959547

ORIGINAL T-Test Method output

```
1 df_sig10
```

	0	1	2	3	4	5	6
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	True	False	True	False	False
One tailed T test, Greater, P Values	True	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	True	False	True	True	True	True	True
...	...	...	...	...	...	...	...
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	False	False	False	False	False
One tailed T test, Greater, P Values	False	False	False	False	False	False	False

192 rows × 7 columns

CHAPTER 11:
BIBLIOGRAPHY

11 Bibliography

1. Alghofaili, Yasuhito Sawahata, Haikun Huang, Hsueh-Cheng Wang, Takaaki Shiratori, and Lap-Fai Yu. 2019. Lost in Style: Gaze-driven Adaptive Aid for VR Navigation. In CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK. ACM, New York, NY, USA 13 Pages. https://doi.org/10.1145/3290605.3300578
2. Anderson, N. C., Bischof, W. F., Laidlaw, K. E. W., Risko, E. F., & Kingstone, A. (2013). Recurrence quantification analysis of eye movements. *Behavior Research Methods*, *45*(3), 842–856. https://doi.org/10.3758/s13428-012-0299-5
3. Bornstein, M. H. (1980). [Review of *The Ecological Approach to Visual Perception*, by J. J. Gibson]. *The Journal of Aesthetics and Art Criticism*, *39*(2), 203–206. https://doi.org/10.2307/429816
4. Carpenter, S. (2001). *Sights unseen*. Monitor on Psychology. 32(4), p54. https://www.apa.org/monitor/apr01/blindness
5. Carrasco, M. (2011). Visual attention: The past 25 years. *Vision Research*, *51*(13), 1484–1525. https://doi.org/10.1016/j.visres.2011.04.012
6. Childs, H., Brugger, E., Whitlock, B., Meredith, J., Ahern, S., Pugmire, D., Biagas, K., Miller, M.C., Harrison, C., Weber, G.H., Krishnan, H., Fogal, T., Sanderson, A., Garth, C., Bethel, E.W., Camp, D., Rubel, O., Durant, M., Favre, J.M., Navratil, P. (2012). "VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data". In "High Performance Visualization - Enabling Extreme-Scale Scientific Insight". Chapman and Hall (CRC). First Edition, Ch.16, 358-396, https://doi.org/10.1201/b12985
7. Cinz. (2021). Man in Future Living Metaverse [Vector]. PNGTree. https://pngtree.com/freepng/man-in-future-living-metaverse_6960080.html
8. Clay, V., König, P., & König, S. U. (2019). Eye tracking in virtual reality. *Journal of Eye Movement Research*, *12*(1). https://doi.org/10.16910/jemr.12.1.3
9. Cleveland, D. (2017). Theory, Practice, and Standardization of Eye-tracking Technology. Google TechTalk. https://www.youtube.com/watch?v=wi19uS4JFJ4&t=746s
10. David, E., Beitner, J., & Võ, M. L.-H. (2020). Effects of Transient Loss of Vision on Head and Eye Movements during Visual Search in a Virtual Environment. *Brain Sciences*, *10*(11), 841. https://doi.org/10.3390/brainsci10110841
11. David, E. J., Beitner, J., & Võ, M. L.-H. (2021a). The importance of peripheral vision when searching 3D real-world scenes: A gaze-contingent study in virtual reality. *Journal of Vision*, *21*(7), 3. https://doi.org/10.1167/jov.21.7.3
12. De Brigard, F., & Prinz, J. (2010). Attention and consciousness. Wiley interdisciplinary reviews. Cognitive science, 1(1), 51–59. https://doi.org/10.1002/wcs.27
13. Drewes, H. (2010). *Eye Gaze Tracking for Human-Computer Interaction*. Dissertation. München. https://edoc.ub.uni-muenchen.de/11591/1/Drewes_Heiko.pdf
14. Drewes, J., Feder, S., & Einhäuser, W. (2021). Gaze During Locomotion in Virtual Reality and the Real World. *Frontiers in Neuroscience*, *15*, 656913. https://doi.org/10.3389/fnins.2021.656913
15. Enders, L. R., Smith, R. J., Gordon, S. M., Ries, A. J., & Touryan, J. (2021). Gaze Behavior During Navigation and Visual Search of an Open-World Virtual Environment. *Frontiers in Psychology*, *12*, 681042. https://doi.org/10.3389/fpsyg.2021.681042

16. Feldmann-Wüstefeld, T., Brandhofer, R., & Schubö, A. (2016). Rewarded visual items capture attention only in heterogeneous contexts. *Psychophysiology*, *53*(7), 1063–1073. https://doi.org/10.1111/psyp.12641
17. Findlay, J. M. (2009). Saccadic eye movement programming: Sensory and attentional factors. *Psychological Research*, *73*(2), 127–135. https://doi.org/10.1007/s00426-008-0201-3
18. Fridman, L., Reimer, B., Mehler, B., & Freeman, W. T. (2018). Cognitive Load Estimation in the Wild. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 1–9. https://doi.org/10.1145/3173574.3174226
19. Gibson, J. J. (1977). The theory of affordances. *Hilldale, USA*, *1*(2), 67-82.
20. Gillies, M., Fiebrink, R., Tanaka, A., Garcia, J., Bevilacqua, F., Heloir, A., Nunnari, F., Mackay, W., Amershi, S., Lee, B., d'Alessandro, N., Tilmanne, J., Kulesza, T., & Caramiaux, B. (2016). Human-Centred Machine Learning. *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, 3558–3565. https://doi.org/10.1145/2851581.2856492
21. *Glaholt, M.G. (2014). Eye Tracking in the Cockpit: a Review of the Relationships between Eye Movements and the Aviators Cognitive State.* Defense Research and Development Canada.
22. Gogel, W. C. (1990). A theory of phenomenal geometry and its applications. *Perception & Psychophysics*, *48*(2), 105–123. https://doi.org/10.3758/BF03207077.
23. Hayashi, M., Oman, C.M., & Zuchlag, M. (2003). Hidden Markov Models as a Tool to Measure Pilot Attention Switching during Simulated ILS Approaches. International Symposium on Aviation Psychology, 502-507, Dayton OH, Apr. 14-17. http://hdl.handle.net/1721.1/28912.
24. Hommel, B., Chapman, C. S., Cisek, P., Neyedli, H. F., Song, J.-H., & Welsh, T. N. (2019). No one knows what attention is. *Attention, Perception, & Psychophysics*, *81*(7), 2288–2303. https://doi.org/10.3758/s13414-019-01846-w
25. Imaoka, Y., Flury, A., & De Bruin, E. D. (2020). Assessing Saccadic Eye Movements With Head-Mounted Display Virtual Reality Technology. *Frontiers in Psychiatry*, *11*, 572938. https://doi.org/10.3389/fpsyt.2020.572938
26. Intriligator, J., & Cavanagh, P. (2001). The Spatial Resolution of Visual Attention. *Cognitive Psychology*, *43*(3), 171–216. https://doi.org/10.1006/cogp.2001.0755
27. Koch, C. (2018). *What is Consciousness*? https://www.nature.com/articles/d41586-018-05097-x
28. Lawrence Livermore National Laboratory. (2022) VisIt. [Version 3.3.3] Lawrence Livermore Labs. [Computer Software] https://visit-dav.github.io/visit-website/
29. Long, X. (2010). *Eye Movement Tracking for Diagnostic Systems*. Dissertation. https://api.semanticscholar.org/CorpusID:114359466
30. Loschky, L. C., Szaffarczyk, S., Beugnet, C., Young, M. E., & Boucart, M. (2019). The contributions of central and peripheral vision to scene- gist recognition with a 180 visual field. *Journal of Vision*. 19(15) doi:https://doi.org/10.1167/19.5.15
31. Louw, T., & Merat, N. (2017a). Are you in the loop? Using gaze dispersion to understand driver visual attention during vehicle automation. *Transportation Research Part C: Emerging Technologies*, *76*, 35–50. https://doi.org/10.1016/j.trc.2017.01.001
32. Matin, Ethel. (1974). Saccadic suppression: A review and an analysis. Psychological bulletin. 81. 899-917. 10.1037/h0037368.
33. Miseviciute, I (2023). Eye Movement, Types and functions explained. Tobii Learn Article. https://www.tobii.com/resource-center/learn-articles/types-of-eye-movements

34. Mon-Williams M, Plooy A, Burgess-Limerick R, Wann J. Gaze angle: a possible mechanism of visual stress in virtual reality headsets. Ergonomics. 1998 Mar;41(3):280-5. Doi: 10.1080/001401398187035. PMID: 9520625.
35. Nakayama, K., & Loomis, J. M. (1974). Optical Velocity Patterns, Velocity-Sensitive Neurons, and Space Perception: A Hypothesis. *Perception*, *3*(1), 63–80. https://doi.org/10.1068/p030063
36. Nilsson, E. J., Aust, M. L., Engström, J., Svanberg, B., & Lindén, P. (2018). Effects of cognitive load on response time in an unexpected lead vehicle braking scenario and the detection response task (DRT). *Transportation Research Part F: Traffic Psychology and Behaviour*, *59*, 463–474. https://doi.org/10.1016/j.trf.2018.09.026
37. Parr, T. Lapusan, T. Grover, P. (2021) DTreeViz. [Computer Software] MIT Licence. Retrieved from https://github.com/parrt/dtreeviz
38. Purves D, Augustine GJ, Fitzpatrick D, et al., editors. Neuroscience. 2nd edition. Sunderland (MA): Sinauer Associates; 2001. Types of Eye Movements and Their Functions. Available from: https://www.ncbi.nlm.nih.gov/books/NBK10991/
39. Qin, H., Zhou, X., Ou, X., Liu, Y., & Xue, C. (2021). Detection of mental fatigue state using heart rate variability and eye metrics during simulated flight. *Human Factors and Ergonomics in Manufacturing & Service Industries*, *31*. https://doi.org/10.1002/hfm.20927
40. Rucci, M., Iovin, R., Poletti, M. Santini F. Miniature eye movements enhance fine spatial detail. Nature 447**,** 852–855 (2007). https://doi.org/10.1038/nature05866
41. Runeson, S. (1977). On the possibility of "smart" perceptual mechanisms. *Scandinavian Journal of Psychology*, *18*(1), 172–179. https://doi.org/10.1111/j.1467-9450.1977.tb00274.x
42. Russel, S. & Norvig, P. (2010) Artificial Intelligence: A modern approach. 3rd edition, Prentice-Hall, Upper Saddle River.
43. Shojaeizadeh, M., Djamasbi, S., Paffenroth, R. C., & Trapp, A. C. (2019). Detecting task demand via an eye tracking machine learning system. *Decision Support Systems*, *116*, 91–101. https://doi.org/10.1016/j.dss.2018.10.012
44. Sipatchin, A., Wahl, S., & Rifai, K. (2021). Eye-Tracking for Clinical Ophthalmology with Virtual Reality (VR): A Case Study of the HTC Vive Pro Eye's Usability. *Healthcare*, *9*(2), 180. https://doi.org/10.3390/healthcare9020180
45. Stoltzfus, J.C. (2011), Logistic Regression: A Brief Primer. Academic Emergency Medicine, 18: 1099-1104. https://doi.org/10.1111/j.1553-2712.2011.01185.x
46. Treisman, A., & Gelade, G. (1980). A feature integration theory of attention. *Cognitive Psychology*, *12*, 97-136.
47. Triantafyllidis, A. K., & Tsanas, A. (2019). Applications of Machine Learning in Real-Life Digital Health Interventions: Review of the Literature. *Journal of Medical Internet Research*, *21*(4), e12286. https://doi.org/10.2196/12286
48. V_ctoria. (2024). AdobeStock. [Images] Room perspective grid backgorund 3d vector illustration. Architecture model projection background template. Line one point perspective horizon perspective scheme. https://stock.adobe.com/images/room-perspective-grid-background-3d-vector-illustration-architecture-model-projection-background-template-line-one-point-perspective-horizon-perspective-sheme/439054635]
49. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (n.d.). *Attention is All you Need*.

50. Vortmann, L. M., Knychalla, J., Annerer-Walcher, S., Benedek, M., & Putze, F. (2021). Imaging Time Series of Eye Tracking Data to Classify Attentional States. *Frontiers in neuroscience*, *15*, 664490. https://doi.org/10.3389/fnins.2021.664490
51. Warren, W., Hannon, D. Direction of self-motion is perceived from optical flow. *Nature* 336, 162–163 (1988). https://doi.org/10.1038/336162a0
52. Williams, L. J. (1988). Tunnel Vision or General Interference? Cognitive Load and Attentional Bias Are Both Important. *The American Journal of Psychology*, *101*(2), 171. https://doi.org/10.2307/1422833
53. WinWin (2024). People Top View. [Vector illustration set]. Adobe Stock. https://stock.adobe.com/search?k=top+of+head+icon&search_type=usertyped&asset_id=413314791
54. Wu, C., Cha, J., Sulek, J., Zhou, T., Sundaram, C. P., Wachs, J., & Yu, D. (2020). Eye-Tracking Metrics Predict Perceived Workload in Robotic Surgical Skills Training. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *62*(8), 1365–1386. https://doi.org/10.1177/0018720819874544
55. Yang, Q. (2017). *The Role of Design in Creating Machine-Learning-Enhanced User Experience*. The AAAI 2017 Spring Symposium on Designing the User Experience of Machine Learning Systems Technical Report SS-17-04. https://www.researchgate.net/publication/315718286_The_Role_of_Design_in_Creating_Machine-Learning-Enhanced_User_Experience
56. Yang, Q. (2021). SIGCHI Outstanding Dissertation Award: Profiling Artificial Intelligence as a Material for User Experience Design. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, 1–3. https://doi.org/10.1145/3411763.3457783
57. Zaman F., Drake W., Intriligator D. J., Gardony D. A. (2021). Investigating a Virtual Reality Based Subterranean Scenario Examining Augmented Reality Implications for Military Operators. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*.
58. Zaman, F. (2022). Dynamic Information Needs Analysis: Understanding User Information Needs in Subterranean Warfare. [Doctoral dissertation, Tufts University]. Tisch Library. https://dl.tufts.edu/concern/pdfs/1544c3855
59. Zaman, F, Roeca G, Lupascu I, Gardony A, Rife J, and Intriligator J. (2021). Analyzing spatial memory in a virtual reality-based subterranean scenario: implication for military augmented reality systems, *Proc. Int. Annual Meeting of the Human Factors and Ergonomics Society (HFES)* [Virtual]. doi: 10.1177/1071181321651183
60. Zaman F, Rife J, Intriligator J, Hannon D. Dynamic Information Needs Analysis: Understanding User Information Needs in Subterranean Warfare. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 2020;64(1):1891-1895. doi:10.1177/1071181320641455
61. Zemblys, R., Niehorster, D. C., Komogortsev, O., & Holmqvist, K. (2018). Using machine learning to detect events in eye-tracking data. *Behavior Research Methods*, *50*(1), 160–181. https://doi.org/10.3758/s13428-017-0860-3
62. Zhao, Q. (2017). *Computational and Cognitive Neuroscience of Vision*. Springer Singapore. https://doi.org/10.1007/978-981-10-0213-7.