Design and development of a performance evaluation framework for remote eye gaze estimation systems

by

Anuradha Kar

College of Engineering and Informatics

National University of Ireland Galway

This dissertation is submitted for the degree of

Doctor of Philosophy

April 2019

Supervisor: Prof. Peter Corcoran

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 80,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Anuradha Kar
April 2019

Acknowledgments

I owe my sincere gratitude to my Ph.D supervisor Prof. Peter Corcoran for his continuous guidance, encouragement and support to my research, for his patience, motivation, optimism, and immense knowledge. He has been the perfect mentor who has inspired me at every step of my PhD journey to pursue excellence through creative ideas, innovation and hard work. He has been there at any time I reached out for help during my research and writing of this thesis. I am truly fortunate to receive his positive influence in my academic career.

I am indebted to Prof. Christopher Dainty for his feedback on my work and providing his valuable insights on eye gaze and optical technologies. I would also like to thank Dr. Petronel Bigioi and Dr. Alexandru Drimbarean for taking me as a part of the FotoNation family and allowing me to use the resources and laboratory facilities of Fotonation. The highly pro-active work environment at Fotonation and the technical expertise of its members had a huge impact on the development of my knowledge and shaping of my research ideas and I will be forever grateful for being able to work on my PhD thesis as a part of this industry-university collaboration program.

I would like to acknowledge Rhys Mulryan, Arpad Zoldi and Pat Fortune, who helped me to set up and run my experiments at Fotonation. I am extremely grateful to all the members of Fotonation who kindly agreed to act as participants in my experiments. Without your warm cooperation, a lot of the developments in this thesis work wouldn’t have been possible. My special thanks would go to Dr. Claudia Costache for her constant help and support throughout my PhD journey.

No words are enough to express my gratitude towards my parents and family in Kolkata, India to whom this PhD thesis work is dedicated. You are my pillars of strength and the wind beneath my wings.

I would like to sincerely acknowledge Science Foundation Ireland for providing me the four year scholarship and generous funding for carrying out my Ph.D research in Ireland.

Abstract

In this dissertation, a comprehensive evaluation framework for remote eye gaze estimation systems that are implemented in consumer electronics applications is developed. For this, firstly, a detailed literature review was made which helped to gain deep insights about the current state-of-the-art in eye gaze estimation algorithms and applications, by categorizing eye gaze research works into different consumer use cases. The wide range of existing gaze estimation algorithms were classified and their applications in interdisciplinary areas such as human computer interactions, cognitive studies and consumer electronics platforms like automotive, handheld devices, augmented and virtual reality were summarised. The review further identified the major challenges faced by contemporary remote gaze estimation systems, which include variable operating conditions such as user distance from tracker, viewing angle, head pose and platform movements that have significant impact on a gaze tracker's performance. Other issues include deficit of common evaluation methodologies, standard metrics or any comprehensive tools or software which may be used for quantitatively evaluating gaze data quality and studying impact of the various challenging operating conditions on gaze estimation accuracy.

Based on the outcomes of this review, the concept of a dedicated performance evaluation framework for remote eye gaze estimation systems was formulated. This framework was implemented in this thesis work through the following steps: a) defining new experimental protocols for collection of data from a remote eye tracker operating under several challenging operating conditions b) collection of gaze data from a number of participants using a commercial remote eye tracker under variable operating conditions c) development of a set of numerical metrics and visualization methods using the collected data to express gaze tracking accuracy in homogeneous units and quantitatively explore gaze data characteristics and quality d) implementing machine learning models using the collected gaze datasets to identify and predict error patterns produced in gaze data by different operating conditions e) development of a software and web-application that incorporates the developed metrics and visualization methods into user-friendly graphical interfaces f) creation of open source code and data repositories containing the performance evaluation tools and methods developed in this thesis, so that they can be used by researchers and engineers working with remote gaze estimation systems.

The aim of this dissertation is to present a set of methods, data, tools and algorithms as analytical resources for the eye gaze research community to use for better understanding of eye tracking data quality, detection of anomalous gaze data and prediction of possible error levels under various operating conditions of an eye tracker. Overall, these methods are envisioned to improve the quality and reliability of eye tracking systems operating under practical and challenging scenarios in current and future consumer applications.

List of figures

Figures in Appendix K

List of tables

Table of Contents

Appendices

Chapter 1

Introduction- Performance evaluation of eye trackers and gaze data quality

Eye gaze research has evolved into a potential avenue for human computer interactions and understanding human attention, interest and cognitive processes [1] [2] since eye gaze provides rich information on human behaviour, affect, intent or even personality traits [3]. A wide range of eye tracking systems and algorithms have been developed till now [4] [5] and the applications of eye gaze changed from mainly clinical uses in the 1990's [6] to growing applications in consumer electronics [7], such as in gaming, virtual or augmented reality and e-commerce to name a few [8] [7] [9].

An eye tracker's accuracy and system behaviour play critical roles in determining the reliability and usability of eye gaze data obtained from them [10][11]. However, in contemporary eye gaze research, there exists a lot of ambiguity in the definitions of gaze estimation accuracy and lack of well-defined methods for evaluating the performance of eye tracking systems [12] [13]. As a result practical accuracy levels of eye trackers may differ significantly from expected values [14] and eye gaze information may lose its applicability in different scientific and consumer use cases [15] [16].

In this dissertation, this critical area of eye gaze research is addressed, i.e. comprehensive performance evaluation of gaze estimation systems and quantitative analysis of gaze data quality through development of experimental methods, data analysis algorithms and relevant software tools. This chapter presents a brief summary of eye gaze uses cases existing in present day consumer electronics as well as describes the fundamentals of gaze estimation techniques used for remote eye trackers- which forms the main eye gaze use case for investigation in this thesis. The major challenges faced by remote eye tracking systems in various consumer applications are highlighted, followed by describing the contributions in this thesis towards addressing these challenges. The thesis structure is discussed and a list of publications made as part of this thesis work that have been done or are currently under review, is presented.

1.1 Eye tracking fundamentals

Eye gaze estimation systems broadly fall in two main categories: the remote or screen based trackers [17] [18] and the wearable or head mounted trackers [19] [20]. The works in this thesis are based on remote eye trackers and all the data collection, analysis and developmental aspects have been done within the scope of these trackers. The reason for focussing on remote eye trackers is the availability of suitable equipment and also due to the relevance with the projects ongoing at the industry partner Fotonation Ltd. A summary of currently existing eye gaze use cases is provided below, followed by description of basic concepts related to remote eye tracking setups, the gaze estimation process and various eye movement measures commonly used in gaze applications.

1.1.1 Eye gaze use cases

Gaze estimation systems can be broadly classified into several use cases, based on their setup and applications in various consumer electronics systems. A literature review paper [12] which was compiled in the beginning of this thesis work gives a more detailed description of these eye gaze use cases. A summary of consumer electronics use cases of gaze tracking systems is provided below.

A. Head mounted systems

This type of gaze trackers requires that the user wears a head mounted device and the tracking setup is very close and in a fixed position relative to the user eye. Such systems can detect gaze points with high accuracy [21] [22]. The general setup includes two cameras; one (eye camera) pointed at the wearer's eye, to detect the pupil; and the other (scene camera) capturing the wearer's point of view, with sometimes additional components like NIR light sources and hot mirrors.[23] [24] Key applications of these setups are in Virtual and Augmented reality (VR and AR respectively) research [25][26]. The main problems in head mounted setups occur from motion and depth of field blur, loss of calibration and errors due to parallax.

B. Remote or screen based gaze tracking systems

This is the type of eye tracker used in this thesis work. These systems use a fixed platform to position the gaze estimation setups, which are generally placed around 1 meter of distance from the users. These systems are typically used for gaze estimation on desktop computers [27] or smart TVs [28]. Applications of eye gaze on desktop systems have been done for computer communication [29], password entry [30], psychoanalysis and for object selection as a substitute for mouse [31] as gaze based pointing has higher speed and accuracy than manual pointing. On large displays and smart TVs eye movements can be used to select and navigate menus, modify display properties, switch channels and understand user interests. Major problems in desktop eye tracking arise from Midas touch [32], user head movements and large errors at high visual angles near corners of the display screen.

C. Handheld gaze estimation systems

In recent times eye tracking has been implemented on mobile devices like smartphones or tablets where they are used to capture user's gaze information as an input modality [33] [34]. Gaze tracking on handheld devices is done using the device front camera, one or more IR light sources and various computer vision algorithms like edge detection, Haar classifiers and ellipse fitting to determine the eye limbus boundaries. Major problems in this platform occur due to user movement, lighting conditions and platform movement. Gaze information is used to access and highlight menu buttons, creation of 3D display effects and gaze based keyboard input, password entry and gaming [9].

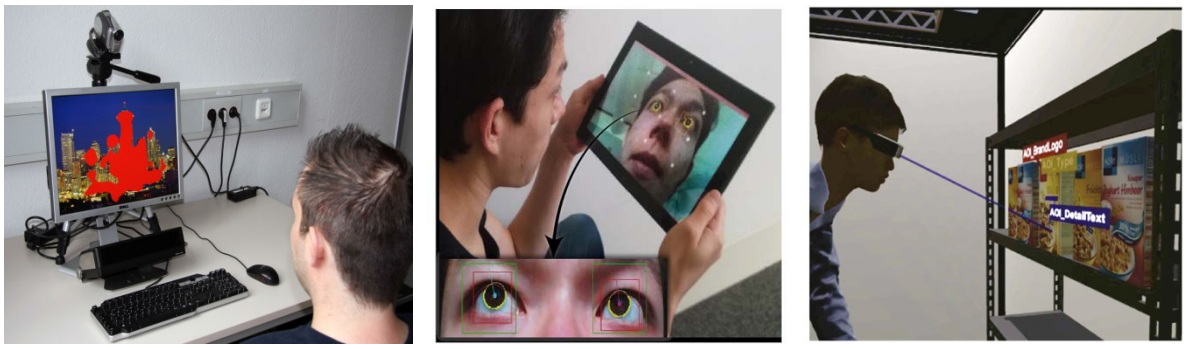


Figure 1.1: Applications of eye tracking in (left to right): desktop, tablet and a large screen TV [13].

D. Eye trackers for automotive environments

In this eye gaze use case the tracking setup is mounted on a highly dynamic platform, e.g. on the dashboard of cars, mainly for monitoring driver behaviour. The setup could be similar to remote eye trackers but operating conditions in automotive use cases are very much different [35][36]. Visual features of the face and eye regions of an automobile driver provide cues about their degree of alertness, perception and vehicle control. Gaze related cues that indicate driver attentiveness include: blink rate, temporal gaze variation, speed of eyelid movements and degree of eye openness [37]. Key issues affecting eye tracking in automotive are problems in eye detection due to variable illumination, occlusion of the eye region due to shadows or eyeglasses, false alarms, and real-time operability [38].

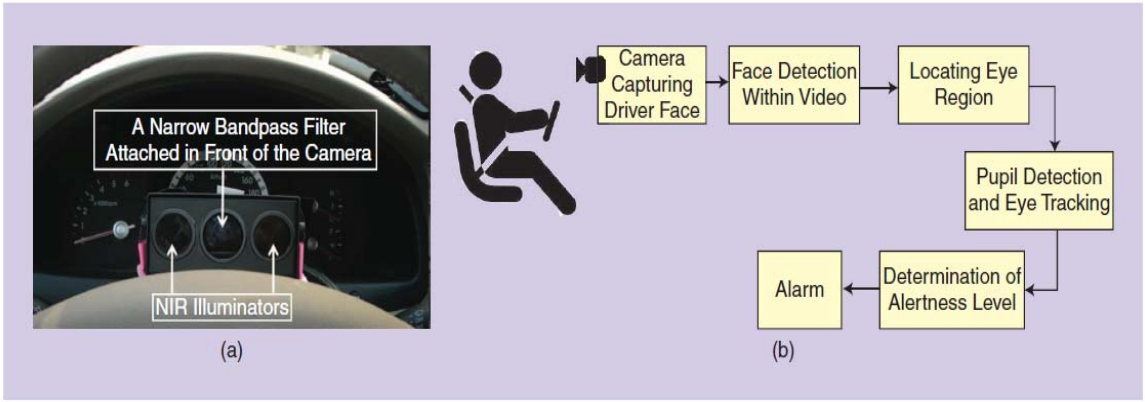
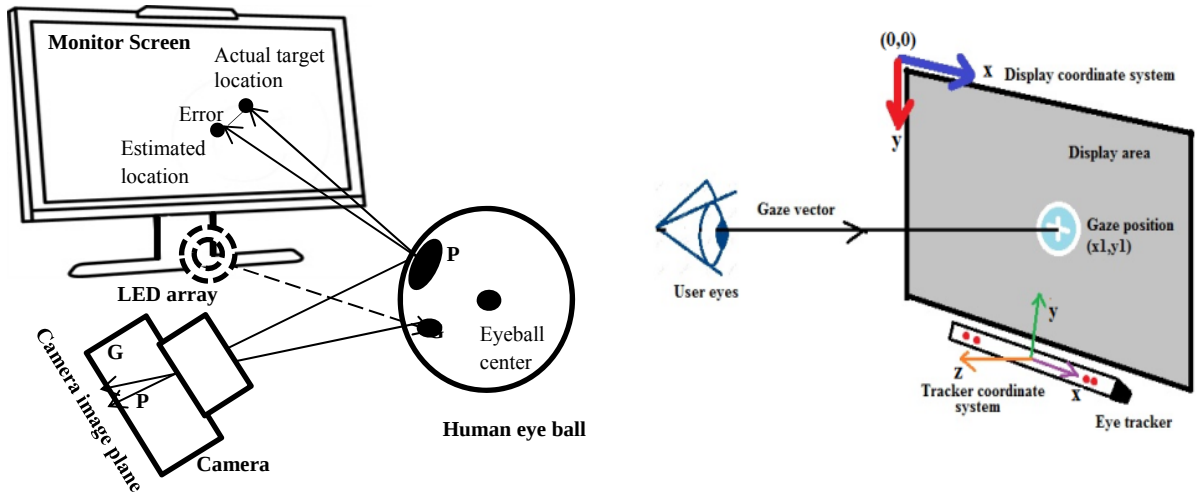


Figure 1.2: (a). Driver monitoring cameras mounted on car dashboard (b) Driver monitoring workflow[9]

1.1.2 General setup for remote eye gaze estimation

Remote eye gaze estimation systems are based on capturing video frames of users' eyes and comprise of one or more digital cameras, near infra-red (NIR) LEDs and a computer with its screen displaying a user interface where a user's gaze is tracked [39]. A typical eye gaze tracking setup is shown in Figure. 1.3a. The steps involved in passive video based eye tracking include user calibration, capturing video frames of the face and eye regions of user, eye detection and mapping with gaze coordinates on the computer screen. Common gaze tracking methods involve using NIR LEDs to produce glints on the eye cornea surface and then capturing images/videos of the eye region [40]. Gaze is estimated from the relative movement between the pupil center and glint positions. Other classes of methods use images of the eye regions for extracting shape and texture features or as direct input to appearance or machine learning models for estimation of gaze directions [7].



(a) (b)

Figure 1.3: (a) Typical setup of a video based remote eye tracking system (b) display and eye tracker coordinate systems

In a typical remote eye tracker setup (as used in this work and shown in Figure 1.3b) the display coordinate system is aligned with the display of the computer and its origin is the upper left corner of the display screen. The eye tracker coordinate system has its origin at the center of the frontal surface of the eye tracker which is aligned with the center of the display screen. The tracker x-axis points horizontally towards the user's right, the y-axis points vertically towards the user's up and the z-axis points towards the user, perpendicular to the front surface of the eye tracker. Gaze data comprises of eye locations of a user tracked by the eye tracker and mapped into the 2D coordinates of the display screen, in units of pixels which may be converted to visual angles if user-tracker distance is known.

Calibration of an eye tracker is a process that helps to accommodate for the differing visual characteristics of one individual from another [41]. This difference arises due to the variable deviation

between the human eye's visual and optical axes (called Kappa angle) from person to person [42]. Calibration is generally done by showing the user a set of specific targets distributed over the display screen of the setup and the user is asked to gaze at them for a certain amount of time [43]. The tracker camera captures the various eye positions of a user for each target which are then mapped to the corresponding gaze coordinates using regression models and thus the tracker learns the mapping function for each person. Usually, calibration-free gaze estimation methods are preferred in consumer electronics use cases for user convenience [44] and such algorithms have been developed using multiple or stereo cameras and 3D eye models or deep learning based methods, such as in [45] [46]. Automated calibration techniques for eye trackers have also been proposed in [47] [48]. Another calibration approach includes participants selecting a stationary marker in their visual field and moving their heads around it in a circular motion, while keeping their gaze fixed at the marker [49].

1.1.3 Overview on eye movements

The common types of eye movements studied in eye gaze research are the following [50]:

1. Fixations: These are phases when the eyes are stationary between movements and visual input occurs. Fixations are used for studying a user's browsing and scene perception characteristics.
2. Saccades: These are rapid and involuntary eye movements that occur between fixations. Saccade related parameters include saccade number, amplitude and fixation-saccade ratio. These are used in gaze based biometrics and understanding visual search processes [51].
3. Scanpath: This includes a series of short fixations and saccades alternating before the eyes reach a target location on screen. Measures derived from scanpath include scanpath direction, duration, length and area covered [52]. Typical uses of scan paths are in assessing quality and layout of user interfaces
4. Gaze duration: It is the sum of all fixations made in an area of interest before the eyes leave that area and the proportion of time spent in each area. It signifies levels of user engagement and interest.
5. Pupil size and blink: These are measures used to study cognitive workload [53].

1.2 Performance evaluation of eye trackers

In a typical eye gaze tracking operation, a user gazes at an interface on a computer screen which provides them with visual stimuli or a set of targets or scene. Gaze tracking accuracy is estimated as the average difference between the real stimuli positions and the measured gaze positions, which also provides an idea about the performance of the system [54]. There are several factors that affect eye tracking accuracy and they are briefly summarized below. The typical issues arising with respect to gaze data quality evaluation in contemporary gaze research are discussed. These lead to the formulation of the research problem addressed in this thesis work.

1.2.1 Practical challenges faced by eye trackers

Through literature reviews and practical works with eye trackers, several parameters were identified in this work, which contribute towards significant increase in gaze estimation errors of remote eye trackers. These factors (also termed as "error sources" in this thesis) frequently affect eye trackers when they operate under practical conditions, such as those found in consumer applications of eye tracking, and are discussed in our papers [55] [56].

The error sources typically affecting a remote eye tracker include:
a) Head pose changes: Random head movement of users lead to distorted appearance of eye-socket geometry in captured eye images, alterations in the user's field of view and for pupil-glint based methods may cause LED glints on the cornea to disappear, leading the tracking algorithm to fail [57].
b) Visual eccentricities: This includes effect of the size of the screen where gaze is tracked and distance of the user from setup. It has been experimentally proven that eye tracking accuracy worsens at the far corners of the display screen due to high visual eccentricities (or visual angles) at those positions [21]. Also if the user is too close to the eye tracker, errors increase as gaze angles are large
c) Platform movements: Typically observed in high dynamic platforms like automotive and handheld devices, variable position, orientation and jitter in the eye tracking setup can disrupt gaze tracking [58]. High frequency eye trackers e.g. operating at 120 Hz are more are also very sensitive to dislocations of the eye cameras and may report large errors due to slight vibrations [48].
d) Changes in illumination: This affects eye feature detection, causes additional glints to appear on cornea surface and may cause an eye tracking algorithm to fail.
e) Human eye limitations-the eyes can fixate as accurately as 10 minutes of visual angle (0.16 degree) which sets the accuracy limit of gaze tracking [59]. High frequency eye movements lead to motion blur and missing gaze points if the frame-rate of the gaze estimation camera is less than 100 Hz [60].

1.2.2 Eye gaze data quality

In contemporary gaze research, data quality refers to the validity of the gaze data measured and reported by an eye tracker [10]. The most common methods of representing gaze data quality are accuracy, latency and precision [11]. Accuracy refers to the difference between the true and the measured gaze direction and may be classified into spatial or temporal accuracy, depending on the deviation between the actual and the measured gaze direction on a sample to sample basis. An eye tracker's data is reported as invalid when relevant eye features could not be detected from the video feed of the eye, e.g. due to loss of the eye image, blurring or occlusion. When the eye tracker reports a valid sample, spatial accuracy can be defined as the deviation (in visual degrees) between the actual

and the measured gaze position of a sample [61]. The difference between the time of actual movement of the eye and time reported by the eye tracker is known as the latency or temporal accuracy. Spatial and temporal accuracy measures along with precision determine data quality from an eye tracker.

There is an important distinction between accuracy and precision. Under repeated measurements of gaze, while accuracy is defined as the mean difference between the measured and true gaze positions, precision, also called reproducibility or repeatability, is the degree to which the repeated measurement of a set of true values produces the same or similar set of measured values regardless of the accuracy of these values. Consequently, measurement can be accurate but not precise and vice versa. If both accuracy and precision differences are zero, the data quality for a gaze sample is said to be optimal. Figure 1.4(b) below denotes the different between accuracy and precision with respect to gaze data.

Another terminology often used in the context of eye tracking data quality is data loss which refers to samples that are reported as invalid by the eye tracker and reported as samples with (0; 0) values or Not a Number (NaN) values. As mentioned above, data losses occur when features in the eye image, e.g. the pupil or the NIR corneal reflections cannot be reliably detected or tracked due to presence of glasses, eyelashes etc. that prevent the eye tracker camera from capturing a clear image of the eye.

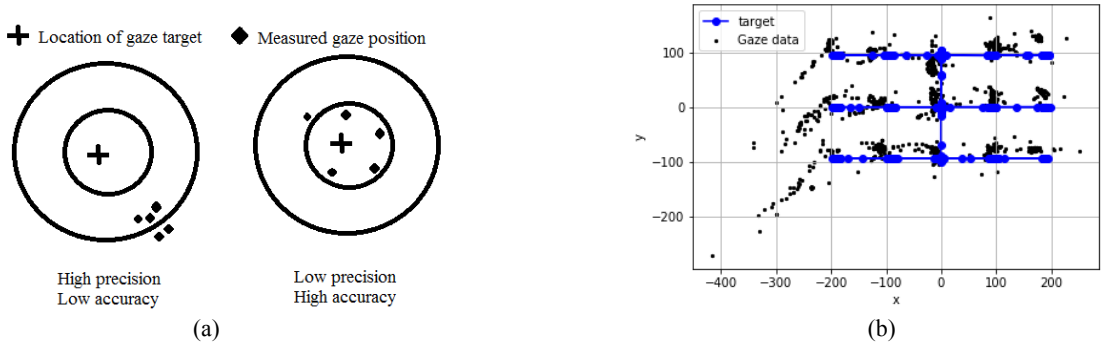


Figure 1.4: (a) Example of accuracy and precision with respect to gaze data (b) An example of poor data quality in which tracked gaze locations (black) have large deviations with respect to target locations (blue).

1.2.3 Current challenges in the evaluation of eye trackers and gaze data quality

From the detailed literature review done in the beginning of this PhD research work [12] several critical issues currently existing in eye gaze research were identified. Firstly, it was found that in conventional eye gaze literature, gaze estimation accuracy measures are presented in different ways e.g. as angular accuracy in degrees, distance accuracy in centi/millimeter or distances in pixels or even gaze detection rates (in percentage). Table 1 below shows the statistics derived from this literature review, surveying nearly 200 articles on gaze research and applications, which highlights the current diversity in metrics used for representing gaze estimation performance. It can be seen from the table that while the commonly used measure is angular resolution (in degrees), other metrics are used

frequently. These gaze data accuracy metrics are not interrelated and sometimes not clearly defined. Also, metrics like gaze detection rates are difficult to interpret physically and such variety in reporting formats makes inter-comparisons between different gaze estimation systems and algorithms impossible. Thus, there exist considerable ambiguities in evaluating performances of gaze tracking systems and specifications of gaze data quality in a homogeneous and standardized way.

Table 1.1 Diversity in performance evaluation metrics used in eye gaze research articles [12]

Eye tracking platform	No. of surveyed papers	No of papers with metric: Degree	No. of papers with metric: Percentage	No. of papers with metric: Others (e.g. pixels, mm)
Desktop	69	44	16	9
Handheld	21	3	9	9
Automotive	35	11	14	10
Head-mounted	57	37	2	18

Another sparsely investigated issue in gaze research is about studying the impacts of different error sources (i.e., system and user variables) on gaze tracking performance [12] which are discussed above in Section 1.2.1. Remote eye trackers, during their operations encounter several variable operating conditions, which are not well described in conventional research or product literature. In desktop use cases, the factors which possibly influence a tracker and its gaze estimation accuracy include user distance, display properties of the screen where gaze is tracked and user head pose variations. On other use cases like handheld devices, platform orientation variations may greatly alter the claimed accuracy of an eye tracker during practical operations. At present, research works commonly do not report impact of these factors, neither are there any dedicated metrics for quantitatively evaluating their impact on gaze estimation accuracy. However in their presence, gaze accuracy may differ significantly from expected values, as was investigated in our experimental works and reported in [13]. Due to the lack of evaluation of these factors, the quality and reliability of gaze data becomes questionable during practical operations in different consumer use cases.

The major challenges in gaze research that are identified and addressed in this thesis are therefore: 1) lack of well-defined and standardized metrics for depiction of gaze estimation accuracy and gaze data quality 2) limited studies on the impacts of non-ideal operating conditions on gaze estimation accuracy 3) lack of any research on analysing gaze error patterns produced by different operating conditions, such as head pose, user distance etc. 4) no existing comprehensive tools or software that can evaluate gaze data quality using data samples from an eye tracker and analysing their spatial and temporal characteristics 5) lack of publicly available gaze datasets that contain the signature of different challenging operating conditions on gaze data.

1.3 Contributions in this Thesis

In this thesis, critical issues existing in contemporary eye gaze research were identified with respect to the lack of standardization of gaze data evaluation metrics and methods for evaluation of eye trackers under the impact of different operating conditions. Also there exist no gaze datasets, tools or software that are dedicated towards studying the accuracy characteristics of eye trackers under variable operating conditions. These formed the basis for proposing the concept of a performance evaluation framework for gaze estimation systems, which would provide suitable methods for in-depth evaluation of gaze data quality and performance of eye trackers, especially those operating under unconstrained conditions in consumer applications.

Development of various components of this evaluation framework has been the focus of this thesis. The main contributions in this thesis are described in Chapters 2-5 and are summarized below. Also the papers published as part of this thesis and linked to the contributions are in Appendices A-F.

1.3.1 Identifying the requirements of standardized metrics and methods for performance evaluation of eye trackers

A detailed literature review was done at the beginning of this thesis work which identified the deficit of standard metrics and evaluation methods for eye trackers implemented in consumer platforms. This review and associated works may be found in [12] [56] [62] which are in Appendix A, B, C and H of this thesis. These papers outline the requirements and concept of the performance evaluation framework for eye tracking systems and form the foundation for later developments in this thesis.

1.3.2 Defining experimental methodology and developing metrics and visualizations to evaluate eye trackers

A set of eye tracking experiments were designed and conducted to collect gaze data and observe the impact of various error sources, such as user distance, head pose, and platform orientations on a remote eye tracker's data. Using the data, a set of fundamental gaze accuracy metrics were derived and several new gaze data metrics and visualizations were developed which may be used to study the impact of multiple operating conditions on a tracker's gaze data quality. These metrics and visualizations are discussed in details in Chapter 4, and are in the paper [13] of Appendix D.

1.3.3 Development of open source software tools for eye tracking data evaluation

The metrics and visualization methods developed in this work are encapsulated into easy to use software and web-application tools for utilization by eye gaze researchers or engineers working in many inter-disciplinary areas that involve eye tracking systems. More details on these developments

may be found in the paper [63] which is in Appendix E, and summarized in Chapter 4. All the software resources related to the methods and tools developed in this work for comprehensive analysis and evaluation of gaze data quality are included to build an open code repository. Details of this repository are in the paper of Appendix G, and summarized in Chapter 5.

1.3.4 Building machine learning algorithms for analysing gaze error patterns

With the data collected from the eye tracking experiments, a labelled dataset containing the signature of several non-ideal operating conditions (or error sources such as head pose, user distance and platform poses) is built. This dataset is then used with supervised classification models to detect the presence of anomalous gaze data and regression models to predict the impact of error sources on gaze data. Details of the machine learning algorithms used for gaze error pattern recognition and modelling may be found in the paper attached in Appendix F and also described in Section 4.4 of Chapter 4.

1.3.5 Creating a benchmark gaze dataset for performance evaluation of eye trackers

A new gaze dataset is created using the labelled gaze data collected during this work which is released into an open data repository with detailed documentation. This data is meant for use by gaze researchers and engineers to compare the performance of their developed gaze estimation algorithms or systems. More details on this are in the paper attached in Appendix G and in Chapter 5.

1.4 Structure of this thesis

This thesis comprises of six chapters including the current one where the research problem and adopted methodologies are discussed.

Chapter 2 presents summaries of several literature reviews made during this thesis work on eye gaze applications, currently existing gaze datasets and recent developments in gaze research. This chapter is linked to the following papers [12] [9] [62] [60] [64].

Chapter 3 describes the concepts and components of the performance evaluation framework for eye tracking systems proposed and implemented in this thesis. This chapter is linked to papers [12] [56].

Chapter 4 includes the implementation details of the performance evaluation framework, describing the developed methods and algorithms. This chapter is linked to papers [13] [63] and papers which are currently under review and may be found in Appendices D, E and F.

Chapter 5 presents the open source tools and data repositories created from the research outcomes of this thesis. This chapter is linked to the paper which is currently under review and in Appendix G.

Chapter 6 discusses the research outcomes of this thesis and potential future works.

1.5 List of Published and "Under Review" Publications

The publications related to this thesis started with an article on eye gaze estimation algorithms and applications in consumer electronics, followed by a detailed review that investigated the issue of performance evaluation existing in contemporary gaze research. The later articles are linked to the development of various components of the performance evaluation framework for eye tracking systems which is proposed in this thesis. Co-authored articles are based on the development of gaze estimation systems for consumer devices using deep learning methods.

1.5.1 Submitted and under review publications

1. A. Kar; P. Corcoran, "Open source code and data repositories for performance evaluation of eye gaze estimation systems", Submitted to Elsevier SoftwareX, March 2019.

2."Eye tracking error classification and modelling with machine learning algorithms", Anuradha Kar, Peter Corcoran, Submitted to SPIE Journal of Electronic Imaging, March 2019

1.5.2 Publications in journals & conferences

1. A. Kar and P. Corcoran, "GazeVisual -a Practical Software Tool and Web Application for Performance Evaluation of Eye Tracking Systems," in *IEEE Transactions on Consumer Electronics*. doi: 10.1109/TCE.2019.2912802

2. J. Lemley, A. Kar, A. Drimbarean and P. Corcoran, "Convolutional Neural Network Implementation for Eye-Gaze Estimation on Low-Quality Consumer Imaging Systems," in *IEEE Transactions on Consumer Electronics*. doi: 10.1109/TCE.2019.2899869.

3. A. Kar; Corcoran, P. Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations. Sensors 2018, 18, 3151.

4. A. Kar and P. Corcoran, "GazeVisual- A Graphical Software Tool for Performance Evaluation of Eye Gaze Estimation Systems," in 2018 IEEE Games, Entertainment, Media Conference (GEM) (2018 IEEE GEM), 2018.

5. J. Lemley, A. Kar, and P. Corcoran, "Eye tracking in augmented spaces: a deep learning approach," in 2018 IEEE Games, Entertainment, Media Conference (GEM). IEEE, 2018, pp. 385–390.

6. A. Kar and P. Corcoran, "A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms," in IEEE Access, vol. 5, pp. 16495-16519, 2017.

7. A. Kar and P. Corcoran, "Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms," 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, 2016, pp. 002061-002066.

8. S. Bazrafkan, A. Kar and C. Costache, "Eye Gaze for Consumer Electronics: Controlling and commanding intelligent systems.," in IEEE Consumer Electronics Magazine, vol. 4, no. 4, pp. 65-71, Oct. 2015. doi: 10.1109/MCE.2015.2464852

9. A. Kar, S. Bazrafkan, C. Costache and P. Corcoran, "Eye-gaze systems - An analysis of error sources and potential accuracy in consumer electronics use cases," 2016 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, 2016, pp. 319-320.

Chapter 2

A review of consumer eye gaze estimation systems

In this chapter, a brief summary of the literature surveys done as part of this thesis (that are linked to various publications) and their main findings are discussed. Supplementary reviews on chronological development of eye tracking technology, gaze research for biometrics, existing eye gaze datasets that have not been included in the publications are presented. Finally, the most recent works published on various eye gaze research areas are summarized and the significance of the developments made in this thesis work towards such gaze applications are highlighted.

This chapter is based on a published literature review paper, an article, a technical journal paper and two published conference papers, the copies of which are in Appendices A, B, H,I,J. These papers are [9] [12] [60] [64] [62].

2.1 Summary of major literature reviews: Eye gaze estimation algorithms, applications and data analysis tools

Several literature reviews were undertaken as part of this dissertation to explore the status of methods used for estimating gaze data quality and identifying sources of errors that are commonly faced by eye tracking systems in different platforms. For this, literature on various gaze estimation algorithms and their applications in consumer platforms like desktop, TV, head-mounted setups and handheld devices were collected and categorized according to: a) the type of consumer electronics use cases in which the gaze estimation systems are applied b)the types of gaze accuracy metrics used in contemporary eye gaze literature c) the types of operating conditions (or error sources) which are typically considered by researchers for evaluating gaze tracking accuracy. A summary of these error sources and use cases are discussed in Chapter 1 and may be found in [12] [9] [62].

The main findings of the literature reviews were that eye gaze estimation systems have been used in a variety of consumer applications and there are several factors which affect eye gaze tracking in different consumer platforms, e.g. variable head pose, user distance and platform movement. These conditions significantly affect eye tracking accuracy, and produce large and unpredictable gaze error

magnitudes. But it was also found that most of the time gaze researchers or designers of gaze based systems do not quantitatively evaluate or specify the impact of these unconstrained operating conditions on their gaze data quality. Other critical issues in current eye gaze research include lack of standard metrics or proper visualization methods for exploring gaze accuracy characteristics and deficit of gaze datasets dedicated towards benchmark comparison and analysis of gaze data quality from different eye tracking systems. Such issues pose serious questions towards the reliability and usability of gaze information in consumer applications.

A survey of existing commercial and open-source software tools for eye trackers was done (included in the paper of Appendix D) which showed that most of the software developed so far aim towards exploration of eye movement characteristics (detecting fixations, scanpath, saccades, eye movement speed, direction, duration), studying eye movement relationships with human behaviour, e.g. building attention maps, deriving regions and sequence of interests and analysing cognitive processes. No dedicated software was found which could be used for quantitative evaluation of gaze data quality, or exploration of gaze data characteristics under variable operating conditions.

These facts learnt from the literature reviews motivated the different research works undertaken as part of this thesis, which include development of metrics, visualization methods, gaze datasets and data analysis software for quantitative evaluation of gaze data quality and gaze estimation accuracy.

2.2 Supplementary review 1: Chronological developments of eye tracking technologies

2.2.1 Scleral search coils and Electro-occulography

Historically, research on eye gaze tracking dates back to the early 1900s starting with intrusive eye tracking techniques like electro-occulography and scleral search coils (Figure 2.2a). The later measures eye movement via coils embedded into a fitted contact lens or a rubber ring that adheres to the eye surface [65] [66]. With magnets placed around the eye, electric currents are generated in the search coils and by measuring the variations in polarity and amplitude of the current the angular displacement and position of the eye may be determined. Electro-occulography utilizes the existence of an electrical potential difference between the cornea and the fundus of the eye (~ 1mV) which varies with the eye position [67]. By placing electrodes near the eye region and measuring these potentials, horizontal and vertical eye movements may be recorded. These methods suffer due to their invasive or intrusive nature, influence by metabolic changes in the eye, susceptibility to changes in skin resistance, surrounding luminance and artefacts due to eyelid and head movement.

2.2.2 Passive video based eye tracking for consumer devices

Remote video-based eye trackers are the most common systems used in consumer applications which could be placed at a specific distance from the user and gaze estimation is achieved by capturing and

processing images of the full face or eye region in natural light or using NIR illumination [68]. The other class of video based eye trackers are head mounted systems which are portable platforms, generally including two cameras [69]. One is the eye camera pointed at the wearer's eye, to detect the pupil; and the other (scene camera) captures the wearer's point of view. These come across as attachment-free, mobile, lightweight devices with simple hardware and are known to provide high accuracy gaze information.

2.2.3 Current generation of eye tracking: The "invisible" eye tracker

After a few decades of development in eye tracking technologies, a very recently released eye tracker named "Invisible" (Figure 2.1b) claims to require no setup, adjustments or calibration and perform robust eye tracking in outdoor environments, where traditional eye trackers have high failure rates. Gaze estimation in this device is implemented using deep neural networks rather than pupil and glint imaging as described above. The eye cameras are completely outside of the field of view of the users and the device can provide provide scene video and binocular gaze data at 200Hz (compared to traditional commercial remote trackers working at 30/60Hz). The device can compensate for movement of the glasses to deliver high quality eye tracking data.

Figure 2.1: (a) A scleral search coil[66] (b) The "Invisible" eye tracker released in January 2019[70]

2.3 Supplementary review2: Review of applications of eye gaze in biometrics

This is a supplemental literature review that was done to study the prospects of using gaze information in biometric studies. This review also helped to learn in great details about the significance of different eye movement features and how to detect them in collected gaze data.

Extraction of individual-specific characteristics from eye movements has led to the development of the field called "eye movement based biometrics" where eye movement patterns are derived and used for person recognition purposes like other biometrics. Inter-person differences in the variability of eye movements occur due to (i) physical oculomotor structure of every person and (ii) the brain activity related to vision based cognition and perception that varies from person to person. A person who is being identified is asked follow certain stimuli on the computer screen and an eye tracking system is used to collect information about eye movements (e.g. saccades) during the process that represents a subject's eye physiology as well as perception behaviour while following the stimulus.

Eye movement biometrics uses the inter-person dissimilarities in eye movements and deals with the formation, comparison, and classification of the respective biometric templates. There are several notable advantages of this biometric such as resistance to shoulder surfing and spoofing resistance because exact replication of saccades is virtually impossible. Also it allows unnoticeable continuous authentication of a person and the identity of the user can be re-authenticated continuously without any interaction with the user.

However, till now the reliability of eye movement biometrics is still not comparable to traditional biometric recognition approaches. Most eye-trackers cannot measure the microscopic movements of the eye – the tremor, drift, and micro-saccades, due to accuracy and temporal resolution limitations. High frequency eye trackers with ~250 Hz sampling frequency are required. The design of stimulus for the person identification has to be done carefully so that it doesn't take too long to capture the eye movements as well there is no learning effect due to viewing the same stimulus. It is also unpredictable how the eye movement patterns may change due to age or illness.

The work in [59] describes the different measures that can be used for studying inter-person variability using eye movements. Eye movements like saccades, pro-saccades, anti-saccades and smooth pursuit are used to derive different measures. Primary pro-saccade measures e.g. the latency of a saccade and the relationship between the amplitude and the peak velocity of the saccade, called the main sequence measure shows the greatest reliability. The paper in [71] studies saccadic movements and associated pre- and post-saccadic lens oscillations (PSO) for differentiation of human subjects. These include saccade duration, amplitude, variance, average speed and average acceleration etc. Saccade features produce a high 86.1 % accuracy while the combination with oscillation features results in 86.4 % accuracy. In [72] both identification and verification is studied based on saccadic eye movement signal measurements and identification rates are 80-90% at their best. Eye movement measures like latency, duration, time to maximum velocity, saccade amplitude, maximum angular velocity, acceleration and deceleration were computed. The work in [73] uses a head-mounted infrared eye tracker with sampling rate of 250 Hz to collect data from 47 subjects who followed a jumping stimulation point presented on a normal computer screen at twelve successive point positions. Average recognition rate was found to be about 7 % to 8 %. In [74] several processing steps were used to extract dynamic features such as the saccadic velocity and acceleration and finally a classification scheme is applied for the identification of the test samples to the subjects. Local eye movement features are derived to represent local differences of the captured gaze time-series data, and used jointly with the Mel Frequency Cepstrum Coefficients(MFCC- a well-known method, used in speech recognition), which is based on the local phase information of eye movement data. The person classification rate improves from 61% to 82% on using this local feature.

2.4 Supplementary review 3: A review on eye gaze datasets for building and testing gaze estimation methods

A wide variety of eye gaze datasets currently exist, which cater to individual research problems and researchers are constantly introducing new datasets. Many datasets have been made publicly available but as eye-region data is a personal biometric some are built and maintained within a specific research group and access is limited due to legal reasons or due to licensing restrictions.

A survey of gaze datasets was made (and presented below) by including publicly available ones and also datasets which have not shared publicly. Several details regarding these datasets were noted- e.g. setup, number of participating members, information about parameters other than gaze included in a dataset- e.g. head pose, eye movements, user distance, data-type (images/video) etc. It was found that gaze datasets can be broadly classified into two types: the ones used for building and testing gaze estimation algorithms (Sections 2.4.1 & 2.4.2) and the others that are used for modelling and validating user attention patterns, cognitive processes, saliency models etc. (Section 2.4.3).

2.4.1 Publicly available datasets

A list of publicly available datasets in eye gaze research is tabulated below.

Table 2.1: Publicly available datasets for building gaze estimation algorithms

No.	Dataset Name	Data type	Purpose/Special notes	Description
1	CAVE [75]	images	Data set built to train a detector to sense eye contact in an image using a passive, appearance-based approach. Can be used for gaze estimation. Data includes user wearing glasses. Head pose variation is calibrated.	56 different people (32 male, 24 female), 21 of our subjects were Asian, 19 White, 8 South Asian, 7 Black, and 4 Hispanic or Latino. Subjects ranged from 18 to 36 years of age, and 21 of them wore prescription glasses. 5,880 high-resolution images, each image has a resolution of 5,184 x 3,456 pixels. For each subject, images acquired for each combination of five horizontal head poses
2.	Weidenbacher et al [76]	images	To be used for evaluations of computational methods for head pose and eye gaze estimation Participants with glasses included	Dataset of 20 subjects including faces in various combinations of head pose and eye gaze. 2220 colour images. For each head pose angle –acquired nine different gaze conditions.
3.	McMurrough et al [77]	videos	To be used as a benchmark for Point of Gaze (PoG) detection algorithms No spectacles, free head motion. Landmarks not annotated. Smooth pursuit recorded.	20 human subjects as they looked at predefined positions of a computer display or followed a moving target. Videos recorded as human test subjects followed a set of predefined points of interest on a computer visual display unit.
4.	UT Multiview [78]	images	Training and testing data for appearance-based gaze estimation methods Multiple camera, different gaze directions and views	50 (15 female and 35 male) subjects. 160 gaze directions per person were acquired using 8 cameras. 64, 000 (= 50 subjects ×8 views ×160 gaze directions) eye images; 8000 (= 50 subjects ×160 gaze directions) 3D eye shape models; 1,152, 000 (= 50 subjects ×144 views ×160 gaze directions)
5.	MPII Gaze [79]	images	For appearance-based gaze estimation in the wild. Free head motion, uncontrolled illumination	People:15 participants. Total 213,659 images. 10,848 image samples manually annotated with position of 6 facial landmarks and position of two pupil centres for each of above images.

6.	OMEG: Oulu Multi-Pose Eye Gaze Dataset [80]	images	Evaluating and comparing gaze tracking algorithms.	People: 50 subjects Over 40000 images from 50 subjects. Sequences are captured under fixed and free head poses. Five landmark labels and gaze angles are provided as the ground truth.
7.	MSP Gaze corpus [81]	video	For appearance-based, user-dependent and independent gaze estimators Recording with/ without head movement, different user distance, free head motion, no gaze location data	People: 46 subjects 14 recordings per session
8.	EYEDIAP [82]	video	For the training and evaluation of gaze estimation approaches Changes in ambient and sensing condition and types of targets	People:: 16 people: 12 male and 4 female The dataset was designed to train and evaluate gaze estimation algorithms from RGB and RGB-D data. 94 session recordings, each with different characteristics.
9.	3D mask attack dataset [83]	video	Biometric face spoofing database with eye positions Has manually annotated eye positions	People: 17 subjects 76500 frames of 17 persons, recorded using Kinect for both real-access and spoofing attacks. 5 videos of 300 frames are captured and in each video, the eye-positions are manually labelled.
10.	HPEG [84]	videos	For head pose and eye gaze estimation algorithm testing. Free movement of subjects	People: 10, 2 female and 8 male. 20 color video sequences.
11	Gi4e [85]	images	Evaluate Haar classifiers to test their ability to detect the eyes when they rotate Note: no head pose information, information on illumination.	People: 103 different subjects, males and females. 1236 colour images. For each user 12 images are recorded.

2.4.2 Datasets developed by research groups but not shared publicly

These gaze datasets have been built by researchers for use cases such as desktop and automotive systems but the datasets are not made public.

Table 2.2: Non-public datasets for gaze estimation

No.	Paper citation	Data type	Purpose/Special notes	Description
1.	[86]	image	Use case: Desktop Build eye tracking system with free head motion.	People: 5 subjects 34,000 images. Head rotation range around 25 degrees for yaw 10 degrees for tilt.
2.	[87]	images	Use case: Desktop Build and test low-cost eye tracking system	People: 6 300 images taken under varying lighting conditions, head positions, with complex backgrounds
3.	[88]	images	Use case: Desktop Estimate five gazing directions	People: 15 different test subjects. Facial images with five different gazing directions, Total of 1000 images for each gazing direction.
4.	[89]	image+ video	Use case: Automotive Driver alertness monitoring	People: 5 subjects of different ethnicity and gender
5.	[90]	images	Use case: Automotive Head pose/ gaze direction recognition to control headlamps	People: 6 people Images taken with 30 fps web camera mounted on front side of a driver to capture the facial images.

6.	[91]	images	Use case: Automotive Head pose and gaze zone estimation Note: Illumination, head pose variations and users with glasses considered,	People: 2 databases with 12 and 11 subjects 300 000 frames. Database contains images at daytime and night-time with subjects wearing glasses and sunglasses. Database 1 contains 85 000 images from 12 subjects. Database 2 contains approximately 200 000 frames from 11 subjects.
7.	[92]	videos	Use case: Automotive Driver fatigue behaviour study	People: 10 subjects 1000 video segments collected, each video segment with 30 seconds.
8.	[93]	Eye tracking data	Use case: Automotive Monitor driver's visual and mental distraction	People: 9 participants, 35-55 years old Eye and driving performance data was collected using a Seeing Machines faceLAB eye tracker.
9.	[94]		Use case: Automotive Determine sensitivity of eye movements to in-vehicle conditions	People: 119 subjects Eye movements measured at 60 Hz with Seeing Machines faceLAB, v. 3.0. For another data set, eye movements collected with head-mounted system.

2.4.3 Eye gaze datasets for cognitive studies

These datasets have been developed with users looking at a series of images while their eye movements/images/videos are recorded. The collected eye movement data is then used for building and validating cognitive studies, visual attention patterns, saliency models etc.

Table 2.3: Eye gaze datasets for cognitive studies

No.	Citation	Description
1.	[95]	Video database for computational models of visual attention. 12 video sequences with eye-tracking data. Gaze fixations recorded using a head-mounted eye-tracker 15 participants (2 women and 13 men).
2.	[96]	Benchmark for testing gaze modelling algorithms. Contains fixation coordinates and eye movement trajectories of 29 observers as they viewed 101 natural calibrated images, and 30 000 fixation points.
3.	[97]	Recorded to demonstrate that faces attract significant visual attention while viewing images through free-viewing, search, and memory tasks.
4.	[98]	Comprises of complex photographic images and was used to validate a saliency model predicting interesting image regions. The study concluded that early eye fixations are observed in symmetrical image areas.
5.	[99]	Aims to validate a frequency domain-based saliency detector incorporating scale-space analysis.
6.	[100]	Publicly available, large-scale eye movement database to aid natural image-related visual attention studies. The EFRs are used to validate a supervised saliency model combining top-down/ bottom-up cues.
7.	[101]	Compiled to study how image resolution affects consistency in eye fixations across observers. The study noted that eye fixations are biased towards the image center for all resolutions.
8.	[102]	Contains a repository of eye fixations to study viewing patterns on semantically rich and diverse images, including faces, portraits, indoor/outdoor scenes, and affective content.
9.	[103]	Contains eye movement recordings while viewing natural scenes to validate a visual saliency model based on the principle of maximizing scene information.
10.	[104]	Provides eye-tracking data for reference images from 3 image quality databases to validate the hypothesis that salient image regions should contribute more to objective image quality metrics
11.	[105]	Two subject groups– an active group of 12 subjects performed action recognition, while a second group of 4 subjects free-viewed the videos. Fixation patterns of free and active
12.	USC CRCNS Datasets	Designed to investigate the role of factors such as memory on visual attention in dynamic scenes. Eye movements were recorded as users viewed normal and scrambled video contents. Found at: https://crcns.org/data-sets/eye/eye-1
13.	[106]	Saliency in Context eye tracking dataset, 1000 images with eye-tracking data in 80 image classes.
14.	[107]	74 video sequences of 5 mins each, captured and annotated more than 500,000 frames. The labeling contains drivers' gaze fixations and their temporal integration
15.	[108]	700 images, 5551 segmented objects, eye tracking data

From the above survey, several characteristics of currently available gaze datasets are observed. Apart from gaze coordinates or eye images/video, other eye region features included in the datasets are pupil size, eye corners, iris, blink rate, eye closure, fixation or smooth pursuit. Commonly setups use a single camera, while some use stereo, HD or multi-camera systems, while some use a RGBD camera or Kinect. Number of participants varies from 4 to 119-with common numbers being 20-50. Private datasets developed by individual research groups generally have a smaller number of people.

The size of datasets range from 200 images or sequences to over a 100,000. A lot of datasets include head pose information while some others do not. Some of the datasets use 3D head models, others use markers or gyro sensors to estimate head pose. Some gaze datasets capture "free-head motion" of users, i.e., the exact angular positions of the head are not known, whereas some datasets give details about the angular position of head. Few datasets report head poses at +/-10 degrees interval while some others at +/-5 degrees or at +/-20 degrees interval. Some datasets have facial features like pupils, eye corners, nose tip, and mouth corners annotated, while most others do not. Datasets using 3D head models usually have automatically annotated facial and eye features whereas some datasets have been annotated manually.

Several datasets include variations of different parameters- such as users with/without glasses, change in illumination and background, varying user distance, race, age etc. Some datasets only record 2D eye tracking data but few datasets include a 3D target to record 3D gaze data. Nearly half the number of datasets consists of images, the others of video frames. Some datasets have eye tracking data included with the eye images and videos while most others do not.

Another major inference from this survey is also that currently there exist no eye tracking dataset that contains labelled eye gaze and ground truth data collected under calibrated variations of user distance, head pose or platform movement, or gaze datasets built specifically for performance evaluation of eye trackers. This formed the basis for the data collection experiments and building of a new eye tracking dataset as part of this work, which is described in later chapters of this thesis.

2.5 Recent trends and developments in eye tracking research

Eye gaze research is currently a rapidly progressing field and is finding newer consumer applications every year in fields like human-computer and human-robot interaction, augmented and virtual reality, driver monitoring and hand-held devices. With rapidly expanding fields of applications, the quality and reliability of gaze data forms an even bigger question since each new application brings with it newer challenges in operating conditions and requirements for high accuracy and reliability. Some recent works on gaze applications are summarized below, and the ways in which they could benefit from the quantitative evaluation methods and datasets developed in this thesis work are discussed.

In human–human and human–agent interaction, gaze information has proven to be a substantial cue to inform ongoing interaction status and study the reactions of humans or agents with their interaction partners. In [109], human eye gaze is used to understand the communication of intent during human-human and human robot interaction in a quantitative manner. The authors experiment with humans performing a set of goal oriented actions and recordings of the upper body and eye gaze motion are used to develop a computational model of the human actions. It was found that for certain actions with just eye gaze information 85% of subjects could read the intentions of the other humans correctly. In [110] significant differences are found between the gaze behaviours of individuals with autism and those without. This work investigates how children with autism disorder initiate joint attention with a gaze contingent robot. In this, participants with and without autism are made to follow and direct an avatar's gaze to a series of referent images. Observing pupil diameter and gaze location data, the two participant groups were classified and their different eye-movement behaviours were used to improve child–robot interaction. The paper [111] also describes results from a study which analyses the visual attention of the participants during periods of eye-hand dis-coordination and develop pathways for more intuitive and faster human-robot interaction. [112] describes a multimodal input method for hands-free interaction using gaze and voice inputs to achieve improved web browsing. Gaze information was seen to provide a better experience with interactions that require spatial context, for example the user could intuitively scroll through the desired positions following gaze orientation, whereas with voice, users need to use numerous commands and distractions. The work in [113] investigates how key size reduction of virtual keyboards affects the accuracy and speed performance of text entry in a gaze-controlled and head-controlled keyboard, in an effort to improve design of such systems. The authors in [114] focus on estimating and tracking visual focus of attention which is a prominent conversational cue in multi-party social interactions. This paper proposes a method that exploits the correlation between eye gaze and head movements to simultaneously track gaze and visual focus. The work [115] investigates individual, collaborative and competitive visual search with visualization of search partners' gaze. Participants were instructed to search a grid of Gabor patches while being eye tracked. It was found that early in collaboration trials, searchers rarely fixated the same elements and reaction patterns of individuals vary, compared to when the tasks are performed with partners.

In the gaze applications described above, poor gaze data quality can result is completely wrong inferential results about the relations between human vision and cognitive aspects. Therefore, to ensure the quality of gaze data used in these studies, the evaluation methods which are developed in this thesis could be useful for gaze researchers as they can readily test the noise, scatter or accuracy levels of their gaze data without any extra effort, before embarking on further analysis of the data.

Potentials of gaze information in a VR environment are explored in [116] which demonstrates the effectiveness of gaze guidance and sensitivity of peripheral vision to different stimuli. A novel content delivery method for provision of 360degree video streaming on mobile VR headsets is shown in [117] which relies on eye-gaze information from integrated eye trackers to provide high quality video in the proximity of users' fixations points, while lowering the quality at the periphery of the users' fields of view. This helps to achieve a trade-off between bandwidth consumption and Quality of Experience (QoE) associated VR content presentation. Similar approach was shown by [118] [55] which reviews the factors limiting the resolution off-axis in AR/VR, in particular the impact of the eye lens. Authors in [119] aims to apply eye tracking to determine the mental processes of a test subject and assess their cognitive abilities in a VR environment. A deep-learning based method for eye tracking in augmented spaces was proposed in [60] which also showed the impact of various eye image qualities on the accuracy of the algorithm.

In the field of driver monitoring, high sensitivity and real-time assessment properties of eye tracking and eye movement analysis has been utilized in recent works like [35]. This work is based on the development of a novel automatic system for driver fatigue detection using features like average fixation time and the pupil area with KNN algorithm. In [120], the authors suggest that a driver's gaze patterns appear prior to and correlate with the driving behaviours, which could be useful for driving behaviour prediction. A monitoring and prediction framework is also proposed for driving assistance applications that extracts the gaze information through analysing facial features, with a deep learning architecture. In [121], a probabilistic relationship is established between the orientation and position of driver head and continuous gaze angles through dense prediction with CNNs. The proposed model obtains the gaze regions with 95% accuracy and can estimates driver visual attention.

Eye tracking in dynamic platforms like automotive and VR accompanies a lot of challenges owing to frequent loss of calibration and noisy data due to factors like user head pose variations and platform movement. Therefore gaze data quality needs to be constantly monitored if eye tracking is to be done reliably. One of the works developed in this thesis described in Chapter 4 comprises of a gaze data evaluation software that allows live capture of gaze data samples from an eye tracker and do near real time evaluation of its quality. This tool is released as a ready to use open source software, which could be useful for evaluating data from gaze applications in dynamic platforms described above.

Eye gaze applications for smartphone and tablet platforms are being considered as the future of personalized gaze tracking, as mentioned by [122], provided mobile device manufacturers include gaze tracking on these platforms. Eye gaze on mobile devices can link entertainment and smart home systems for example by providing gaze based interaction with relevant devices via any communication protocol. In [123], user gaze is tracked on a smartphone or tablet to make adjustments

such as screen magnification, orientations, focus and moving the screen contents depending on eye position. Works like [124] describe usage of eye tracking to observe effect of screen size on immersion and derive conclusions. The paper in [125] presents a new method to determine gaze and compensate for the effects of platform orientation on gaze tracking accuracy. Experimental results on a prototype infrared smartphone show that the new method achieves about 1 degree of accuracy compared to 3.5 degrees accuracy when platform orientation is not considered.

Like the VR or automotive cases, handheld devices also represent a highly dynamic user platform for eye tracking where eye tracker orientations may vary due to hand pose changes. In this work, a quantitative study of the impact of tablet orientations on the accuracy of an eye tracker is made which gives a lot of insight into the way platform orientations affect gaze data quality. Also a web-application is developed in this work, which runs on mobile operating systems to evaluate gaze data quality. This web-application could help in the evaluation of eye trackers mounted on tablet devices as described above. Both of these developments are presented in Chapter 4.

A publicly available eye gaze dataset named I2Head was developed recently [126] which contains ground truth data for head pose, gaze and simplified user face models of 12 subjects. A webcam (30 fps) is used along with head pose sensors, and relative position between the user and camera is calibrated. For each user, 8 sessions are recorded under controlled and free head movements on point grids containing 17 and 65 fixations. The dataset videos are provided in MPEG-4 format.

The eye gaze dataset developed in this thesis work could be used by researchers for benchmark testing and comparison of quality of other gaze datasets as well as of other eye tracking systems. The dataset details are presented in Chapter 5.

2.6 Conclusion

In this chapter, a summary of the main literature reviews conducted as part of this thesis work is provided. We discuss the main findings from the reviews and explain how they created a context and rationale for the subsequent research works and contributions of this thesis. Also supplemental literature reviews are presented and discussed in the context of the research contributions of this thesis. The following inferences were drawn from the literature reviews:

• Gaze data quality and performance evaluation of eye trackers are such fields which were found to lack dedicated methodologies and standardized protocols.

• There are no public datasets which contains eye tracking data relevant to performance evaluation of eye trackers or corresponding numerical or statistical algorithms that enable systematic studies of the impact of environment and operating conditions on gaze data quality.

• There is a lack of standardized gaze data analysis software that can help researchers or engineers to examine and compare gaze dataset characteristics and specify data quality in standardized metrics.

• New gaze based consumer applications are being developed every year which mostly work in unconstrained environments and where gaze data quality gets frequently affected by multiple factors.

All these factors, as established through the detailed literature surveys discussed here helped to shape the research questions addressed in this thesis and formed the motivation towards building a range of gaze data quality evaluation algorithms, gaze datasets and software tools for performance evaluation of eye trackers. that are presented in Chapters 3-5 of this thesis.

Chapter 3

Proposing the concept of an evaluation framework for eye tracking systems

This chapter presents the concept of a performance evaluation framework that is intended to enable all round evaluation of generic and commercial remote eye trackers and their data quality. This framework would be beneficial to gaze researchers as well as engineers building gaze tracking systems or applications, for quantitative estimation and prediction of gaze tracking performance and data quality.

This chapter is based on a published literature review paper and a published conference paper, which are in Appendix A and C, [12] [56].

3.1 Rationale and design criteria for developing performance evaluation strategies for gaze estimation systems

Eye gaze estimation is currently a highly interdisciplinary area of research that has received quite a lot of interest from consumer electronics (CE) owing to the easy availability of computing and hardware resources and increasing demands from fields like human computer and robotic interactions[5] [127]. Most of these CE applications operate under several challenging operating conditions (or error sources), which are discussed in Chapter 1 in Section 1.2.1, and these factors affect the accuracy of gaze estimation algorithms and worsen the quality and reliability of gaze data from them. Other critical issues that exist with respect to systematic performance evaluation of eye trackers are highlighted in Section 1.2.3 of Chapter 1 and in the Conclusion section (Section 2.6) of Chapter 2.
Considering all these factors, the development of comprehensive evaluation strategies for gaze tracking systems appears to be necessary for several reasons:

a. to evaluate impact of various error sources on a gaze tracker's system performance
b. to report system performance quantitatively in uniform and quantitative formats
c. visualize, quantify and compare results from different eye tracking systems under different operating conditions
d. to identify the main challenging factors for a certain eye tracking user platform.

We present here the concept of such an evaluation framework that comprises of methods, tools and algorithms, that would provide practical performance estimates of remote gaze tracking systems and adopt a uniform set of accuracy metrics for specifying their performance.

3.2 Concept of a performance evaluation framework for gaze estimation systems

The evaluation framework developed in this work comprises of methods to quantitatively estimate the performance of remote gaze tracking systems, through collection and analysis of gaze data obtained from them. The framework is built around dedicated gaze data collection experiments and development of numerical metrics, visualization tools, benchmark gaze datasets and pattern recognition algorithms. The purpose of these developments is to enable evaluation of the practical capabilities and limits of a given gaze tracking system (especially under the influence of various error sources) and specification of gaze estimation accuracy in standard units (angular resolutions).

The structure of the framework is outlined in Figure 3.1. The advantage of having such a framework is that it can answer critical queries related to gaze estimation system design and performance, e.g.:

a) How does a particular system perform when compared with other systems under certain operating conditions?

Our approach: To address this query, in this work, standard gaze accuracy estimates are derived starting from raw gaze data. Also a set of metrics and visualizations are developed which can be implemented using the raw data from any (remote) eye tracker and system variables of the eye tracking setups. These methods can be used on gaze data collected from two or multiple eye trackers, or using data from the same tracker operating under different operating conditions, for comparison of the data characteristics numerically and visually.

b) Which operating conditions affect the performance in a particular use-case?

Our approach: In this work a remote eye tracker is used to collect gaze data by subjecting it to operate under several challenging conditions (one at a time). Using the developed metrics and visualization methods, the magnitudes of gaze estimation errors caused by each operating condition can be estimated and relative impacts of the different conditions can be compared.

c) How will an eye tracking algorithm designed for one platform perform in another?

Our approach: In this work, two consumer platforms- a desktop and a tablet, are used to collect gaze data under similar operating conditions from the same remote eye tracker. Gaze data characteristics of the two platforms are then analysed with the developed metrics and visualizations for in depth comparison.

In this way, the framework developed in this thesis can answer many of such queries by adopting a unified approach towards gaze data analysis and evaluation.

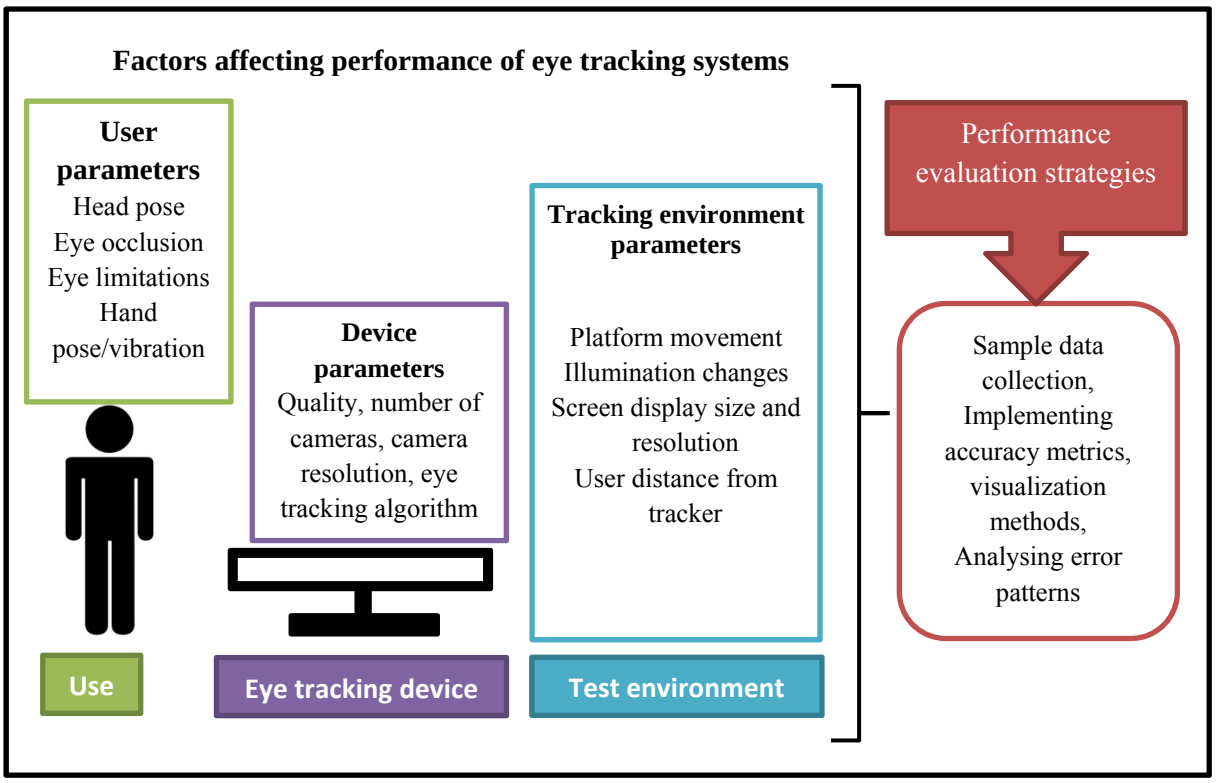


Figure 3.1: An overview of the framework proposed in this work for performance evaluation of eye gaze estimation systems is shown here. Aim of this framework is to test the practical performance limits (accuracy) of a gaze tracking algorithm or system under the influence of various user, system and environmental parameters and specify gaze accuracy in standardized and homogeneous units. The components of the evaluation process include experimental methods to collect sample data from an eye tracker, numerical and statistical algorithms to analyse the collected data and machine learning algorithms to study error patterns in the datasets due to different operating conditions.

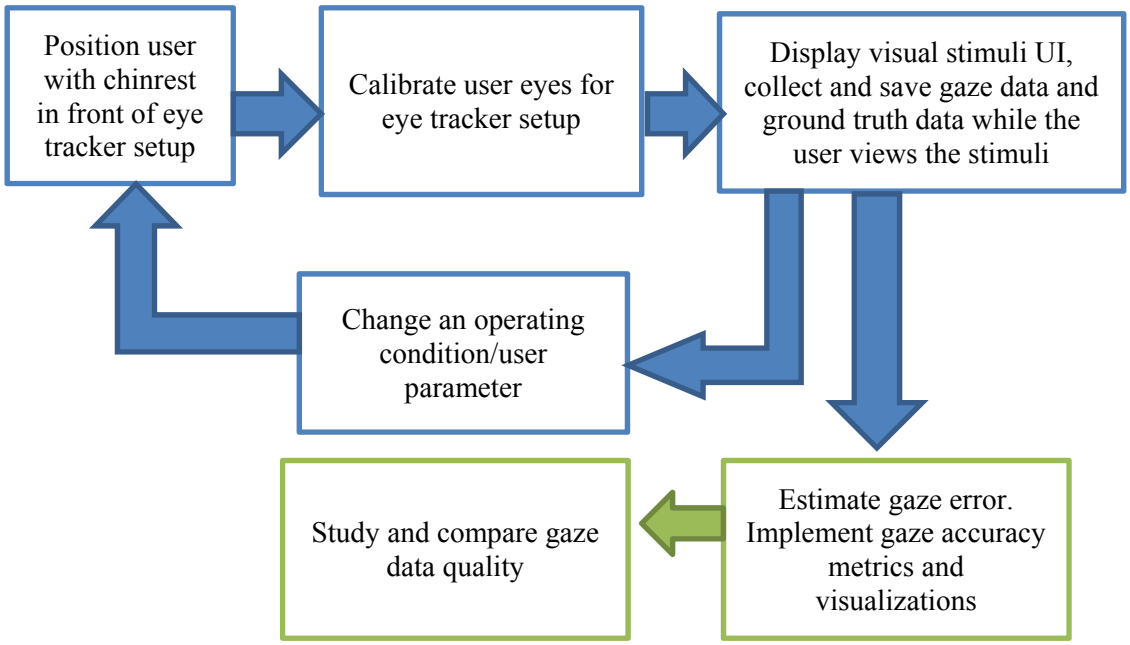


Figure 3.2: Workflow of the performance evaluation framework developed in this work. The variable operating conditions or parameters which are considered in this work are user distance from the tracker, user head pose and platform orientations.

3.3 Components and functioning of the evaluation framework

A schematic and conceptual diagram of the performance evaluation framework is shown in Figure 3.1. A typical eye tracking setup comprises of the user, a gaze tracker mounted on a computer with a display screen and the tracking environment (which varies according to the eye tracking platform, e.g. desktop, automotive etc.). Each of these parts of the setup may influence the performance of the eye tracker by introducing a large number of variable factors which are also listed in Figure 3.1. The proposed evaluation framework takes into consideration a set of these factors and develops eye tracker evaluation methods by collecting and analysing data from an eye tracker under test.
The evaluation workflow is shown in Figure 3.2 in which firstly sample data from the eye tracking system under different operating conditions is collected and secondly, the eye tracking experiments are repeated with a number of participants to collect sufficient data for analysis.

A typical "experiment" or evaluation workflow as shown in Figure 3.2 consists of the following steps:
1) A user is asked to sit in front of the eye tracker setup (discussed in Chapter1), and a certain operating condition (e.g. head pose or user distance) is introduced. User eyes are calibrated for the eye tracker for the current session. The process of eye tracker calibration has been discussed in Chapter 1.
2) The user is presented with a graphical user interface (or UI as in Figure 3.2) which presents visual stimuli to the user. The UI and eye tracker are then started simultaneously. The eye tracker records the gaze coordinates of the user as the user gazes at the UI stimuli points on the screen.
3) The gaze data from the eye tracker is saved. The ground truth data is also collected which comprises of the screen coordinates of UI stimuli points appearing during gaze data collection. The collected gaze and ground truth data are used to estimate gaze error in degrees as the shift between ground truth and tracked gaze locations. The next steps of evaluation are based on analysing the gaze error magnitudes and studying its variability under different conditions.
4)These steps are then repeated with a new operating condition or a new user.

The main components of the evaluation framework developed in this thesis work are the following:

a) new experimental methods for eye tracking data collection that contains signatures of different operating conditions
b) a range of gaze error metrics and visualizations derived from raw gaze data
c) open source tools for analysis and visualization of collected gaze data samples
d) pattern recognition algorithms for detecting and predicting gaze error levels.
e) a new eye gaze dataset created for benchmark testing and evaluation of eye trackers

The components of the framework are shown in Figure 3.3, and discussed in brief below. Details on the implementation of the framework are presented in Chapter 4.

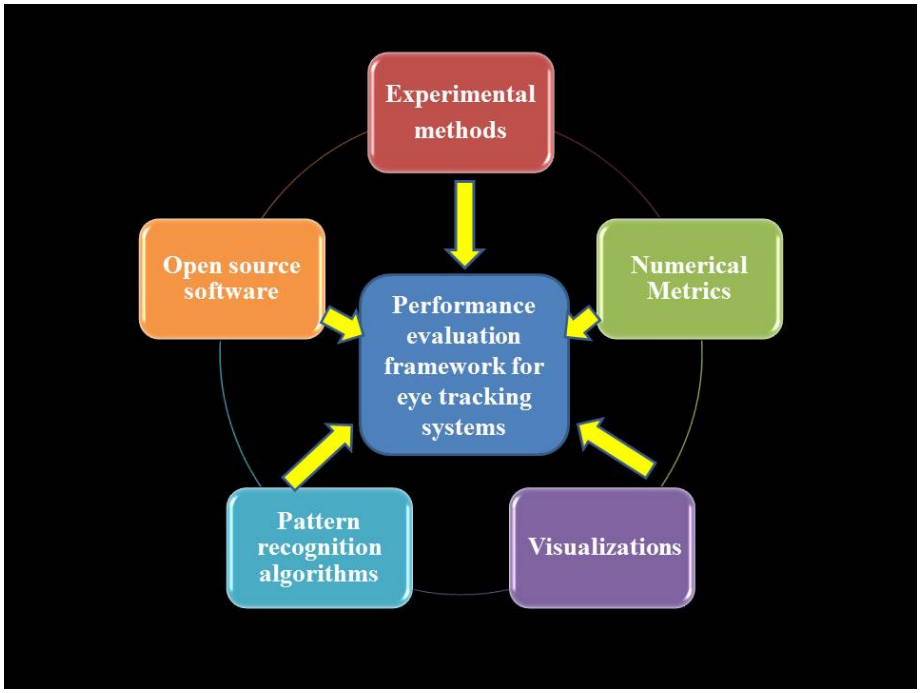


Figure 3.3: Components of the evaluation framework for remote eye gaze estimation systems developed in this thesis.

3.3.1 New experiments for eye tracking data collection

A set of new eye tracking experiments were designed to simulate and observe the impact of various error sources on gaze data from two user platforms, a desktop and a tablet. These experiments are based on studying effects of (i) head pose variations (ii) user distance variations and (iii) platform orientations, (e.g. when eye tracking is done on portable devices like smartphones or tablets). In these experiments a commercial remote eye tracker is used; it works with a Windows desktop and tablet system and has a specified gaze tracking accuracy of 0.5 degree. Table 3.1 provides the list of eye tracking experiments done as part of this thesis for data collection, the operating conditions that were varied during the experiments and the number of subjects for each variation.

Table 3.1: Details of experiments for gaze data collection

Platform	**Desktop**		**Tablet**	
Conditions	User distance	Head pose	User distance	Tablet orientation
Number of variable conditions	4 user distances, 50, 60, 70, 80 cm	6+ 1 neutral= 7 Roll +/- 20 degree Pitch +/- 20 degree Yaw +/- 20 degree	4 user distances, 50, 60, 70, 80 cm	6+ 1 neutral= 7 Roll +/- 20 degree Pitch +/- 20 degree Yaw +/- 20 degree
Number of participants for each variation	20	20	20	20

3.3.2 Developing well-defined gaze-data evaluation metrics

As has been highlighted in the beginning of this chapter, there are definite requirements of methods for quantitative analysis of eye tracking data and characterising an eye tracker's performance under variable operating conditions. As also described in Chapter 1 in Section 1.2.3, the metrics that are currently used to represent eye tracking performance are diverse, not fully defined and also there are no metrics that represent the impact of variable operating conditions on gaze data. Therefore, well defined gaze accuracy metrics and visualization methods are necessary for standardized evaluation of eye tracking performance and gaze data quality.

In this work, the gaze accuracy metrics are derived from basic gaze data coordinates (collected from eye tracking experiments as described above) in standard units of angular resolutions and several other metrics (e.g. sensitivity, spatial density and efficiency) are defined using these gaze accuracy values. Also, visualization methods are developed by graphically representing the gaze error magnitudes to understand and visually compare gaze data quality. With these metrics and visualizations, data from any eye tracker setup may be evaluated and compared in a standardized way and gaze data characteristics can be studied in detail.

3.3.3 Gaze error pattern recognition and modelling algorithms

Raw gaze data doesn't clearly reflect the pattern of gaze errors occurring in a dataset. Even metrics and visualizations may not be sufficient to understand if gaze data is affected by one or more operating conditions or error sources.

To detect error patterns in gaze data caused by different operating conditions, machine learning algorithms are used on the experimentally collected gaze datasets. Gaze data is first cleaned using outlier removal methods, then classifier models such as SVM, K-nearest neighbours and neural networks are used on the datasets to distinguish error patterns, for example, differentiating between errors caused by head pose from errors caused by user distance variations. Along with this, regression algorithms, such as linear, polynomial and neural network based regressors are used to model the error patterns corresponding to the different operating conditions.

The utility of these classification and modelling approaches are that with them, eye gaze researchers can detect the presence of different error sources and predict the possible gaze error levels caused by these. This in turn can help researchers or engineers to design methods to mitigate or correct the impact of these error sources and improve the quality and reliability of eye tracking systems. Details of implementations of these methods are in Chapter 4.

3.3.4 Open source gaze data evaluation tools and labelled gaze datasets

As discussed in Chapter 1 and 2, there are currently no software tools that can be used for the evaluation of gaze data quality. Also there are no labelled gaze datasets representing the impact of different eye tracking conditions on gaze data. Therefore, the gaze data accuracy metrics and visualization methods developed in this thesis work are released into an open source repository for use, modification and upgradation by eye gaze researchers and engineers. Details of this code repository are discussed in Chapter 5.

In addition, all the gaze datasets collected under different operating conditions are made publicly available through an open data repository. This dataset may be used for benchmark comparison of performance of other eye trackers, for example under the influence of varying head poses or user distances or for further research on building advanced gaze data evaluation metrics or understanding error patterns. The contents and details of this dataset and labelling aspects are discussed in Chapter 5.

3.4 Conclusion

The concept and design principles of the performance evaluation framework for eye gaze estimation systems, which forms the core developmental aspect of this thesis are presented here. The components of this framework comprise of experimental protocols, numerical methods, machine learning models and software tools for all round evaluation and characterisation of eye tracking data. This chapter provides an overview of the framework components and the individual components are described with further details in Chapter 4.

Chapter 4

Implementation of the performance evaluation framework

This chapter describes experimental methods which were implemented to collect gaze data from a remote eye tracker while it operated under different operating conditions such as various head poses, user distances and platform orientations. (discussed in Section 4.1). Several new metrics and visualizations for analysing gaze data quality and eye tracker performance were developed using this data. A summary of these metrics and visualizations are provided in Section 4.2 of this chapter.

Data collected from these experiments were used to create a labelled gaze dataset which was then used for training a set of machine learning models to analyse gaze error patterns caused by the different operating conditions mentioned above. Concept and implementation details of the various machine learning algorithms for investigating and predicting gaze error patterns, and the results obtained from these learning algorithms is summarized in Section 4.4 of this Chapter. Details of the labelled gaze dataset are discussed later in Chapter 5.

Apart from these, a software application and its web-application were built as part of this work, which are meant for use by gaze researchers and engineers for quick, easy and quantitative evaluation of gaze data from any remote eye tracker device (description of a generic remote eye tracker setup is described in Chapter 1) or gaze based application. Section 4.3 of this chapter provides details about the developed software and web application. This software, the metrics and visualizations presented in this chapter are released as open resources for use by gaze researchers and engineers for gaze data analysis and evaluation. Details of these open resources are in Chapter 5.

This chapter is based on a published technical paper [13] and a published conference paper [63], and two technical papers which are under review. The copies of the papers are in Appendix D,E, F,G. At several places within this chapter, the relevant sections of these papers are referenced where more background and details may be found regarding the different works discussed in this chapter.

4.1 Summary of experiments for gaze data collection

Details about the gaze data collection experiments done in this work are in the sub-sections below.

4.1.1. Remote eye tracker setup and user platforms for the experiments

For this work, two commercial remote eye trackers were initially selected and tested through pilot gaze data collection experiments to study their individual pros and cons. It was found that out of the two trackers; only one could provide consistent gaze data with sufficient accuracy, and for long user-tracker distances. Therefore, this eye tracker was used for rest of the gaze data collection experiments in this work and its specifications are in Column1of of Table 4.1. More discussions about the different eye trackers used in this work and their gaze data characteristics are presented in sub-section B.

During the experiments, the chosen eye tracker's calibration (as discussed in Chapter 1) was done using the software provided by the tracker's manufacturer which uses 9 static calibration stimuli dots appearing at the corners, top and bottom locations of the display (significance of the eye tracker calibration process is discussed in Chapter 1). After calibration, gaze data collection experiments with the eye tracker are done on two user platforms, a desktop and a tablet. Specifications of the desktop and tablet computer systems used in this work are in Column 4 of Table 4.1. Eye gaze data was collected from the remote eye-tracker while it was mounted on the screen of the desktop or the tablet device and a user was positioned in front of the mounted tracker as shown in Figure 4.2. The head pose of a user was fixed with a chin rest, and user-tracker distance was measured for all experiments.

4.1.2 Eye trackers used in this work

Gaze data collection experiments for this work started with comparing two remote eye trackers, which were Tobii EyeX 4C (Tobii technologies [16]) and the other was Eyetribe [15] (a company currently owned by Occulus VR). Both of these are "bar type" remote screen mountable trackers with specified gaze accuracies of 0.5-1 degree (Figure 4.2). Both the trackers have one or more eye facing cameras and several NIR LEDs. The trackers provide no eye videos and eye camera frame rates of 30Hz are used by both. Operating principles of the trackers are not disclosed by manufacturers.

Pilot data collection was done from both trackers with 10 participants to study and compare their output data quality and characteristics. It was found that both trackers provide left and right eye coordinates and estimated gaze locations (in screen coordinates). For the Tobii tracker gaze data could be obtained for a user-tracker distance range of 50-80 cm but Eyetribe's data quality worsened after user distances of 55-60 cm and it also had connection issues, along with frequent data loss.

Specifications of the remote Tobii tracker which is finally used for all experiments in this work are in Table 4.1. Data comparison of the Tobii and Eye tribe trackers are in Appendix K, where the difference in data quality of these two eye trackers may be clearly seen.

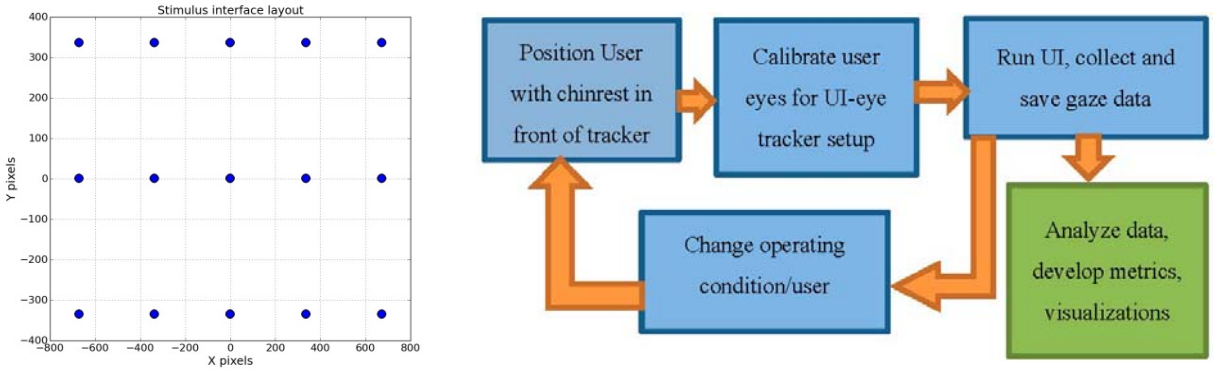
In addition to the remote trackers, a head mounted tracker from Pupil labs (Figure 4.2c) was also used in this work briefly to study its data characteristics. Another motivation was to see if the developed methods and software in this work could be used on data from this tracker as well, since its setup characteristics are very different from the other two remote eye trackers. The Pupil labs tracker [70] is binocular with two eye facing cameras and a world facing camera [24].

It was found that the head mounted Pupil labs eye tracker's output data format is different from that of the remote eye trackers. This tracker provides eye and scene videos, and also gaze data streams in the form of normalized device coordinates [128] which then had to be transformed into the display screen coordinates where gaze was tracked. A discussion on data from this tracker is provided in the paper of Appendix E in its Section V.

The frame rate of the Pupil labs tracker's world camera could be varied between 30-120 Hz and its eye cameras have a frame rate of 200Hz. The tracker's specified gaze accuracy is 0.60 degrees. The major problems faced during experiments with this tracker were frequent loss of calibration due to even slight head movement, requirement for adjustments of eye-camera focus (using in-built focussing ring) for each participant which was quite time-consuming. For these reasons, this tracker was not used for further data collections.

4.1.3 Visual stimuli used for the experiments

Column 2 of Table 4.1 describes the visual stimuli presented to users for data collection [13]in this work. The screen of the desktop or tablet systems displays the visual stimulus interface (also called UI). The UI comprises of a dot moving sequentially on a grid of (5x3) locations (also called areas of interest or AOIs) over the display screen, as shown in Figure 4.1a. The UI dot radius is 10 pixels and it stops at each location for 3 seconds before moving on to the next. The angular extent of the UI stimulus grid is 30 degrees of visual angle at 45 cm distance. The UI runtime is synchronized with a data-logging routine for the eye tracker to collect gaze data while a user gazes at the dots of the UI.



(a) (b)

Figure 4.1: (a) Static view of the UI where gaze is tracked. (b) Flowchart of a single eye tracking experimental session. The on-screen positions traced by the UI dot are known (in screen pixel coordinates).

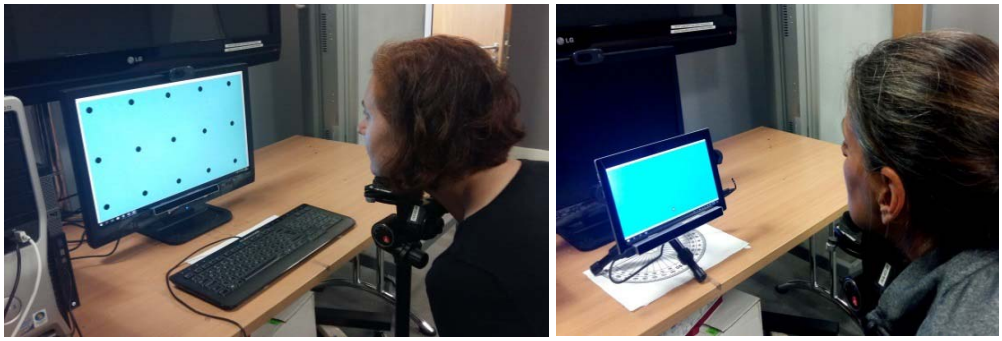
4.1.4 Experimental workflow

The list of experiments done in this work was presented in Table 3.1 of Chapter 3. Eye gaze data is collected from two platforms –desktop and tablet under 4 different user distances, 17 head poses and 6 platform poses. Additional gaze data under for 4 user distances was also collected from a laptop (screen size 14inch, resolution 1366x768 pixels) and is included in the labelled gaze dataset which is an outcome of this work (described in Chapter 5). For all experiments, gaze data from the same set of 20 volunteer participants were collected, in which there were 15 male and 5 female candidates. The participants didn't wear spectacles and the illumination conditions were also same for all experiments.

A typical experimental workflow is shown in Figure 4.1b which was also discussed in Chapter 3 and shown in Figure 3.2. Each gaze data collection session comprised of a user being positioned in front of the eye tracker setup (with desktop or tablet) with a chin rest (Figure 4.2a, 4.2b) and calibrating their eyes. During the data collection session, the UI described above is run on the desktop or tablet screen and users were asked to follow the stimulus dot as it moves. This ensured that the user's fixation distance is closest to stimuli locations. After completion of each session, certain system or user parameter was changed, e.g. registering a new user, varying user distance or head pose and then the experimental sessions are repeated with the new condition.

Table 4.1 : Details of experimental setup with desktop and tablet platforms

Eye Tracker and UI setup	**Details**	**Display and Hardware Characteristics**	**Details**	**Experimental Variables**
Tracker type	Desktop based, NIR LEDs + 1 Camera, 30Hz frame rate	**Screen Size**	Desktop: 22 inch Tablet: 10.1 inch	20 participants for all experiments
Calibration	6 point	**Screen Resolution**	Desktop: 1680 × 1050 Tablet: 1920 × 1200	Fixed and variable user distance
Tolerance	Maximum user distance: 80 cm, spectacles allowed, chin-rest used	**Screen properties**	Desktop: 21.5 inch diagonal, width × height = 18.5" × 11.5" Tablet: 10.1 inch diagonal, width × height = 8.5" × 5.5"	Fixed and variable head pose
User interface	15 AOI locations, AOI radius: 10 pixels	**Pixel sizes of desktop and tablet screens**	Desktop: 0.275 mm Tablet: 0.113 mm	Screen resolution and pixel size
Eye data type	Fixation, AOI duration: 3 s, blinks allowed between AOIs	**Hardware details for desktop and tablet**	Desktop: Core i7, 3.6 GHz, 16 GB ram Tablet: Intel Atom X5, 1.44 GHz,4 GB ram	Platform orientation



(a) (b)

Figure 4.2: Gaze data collection setup showing eye tracker and UI on (a) desktop (b) tablet.

4.1.5 Description of experiments and operating conditions

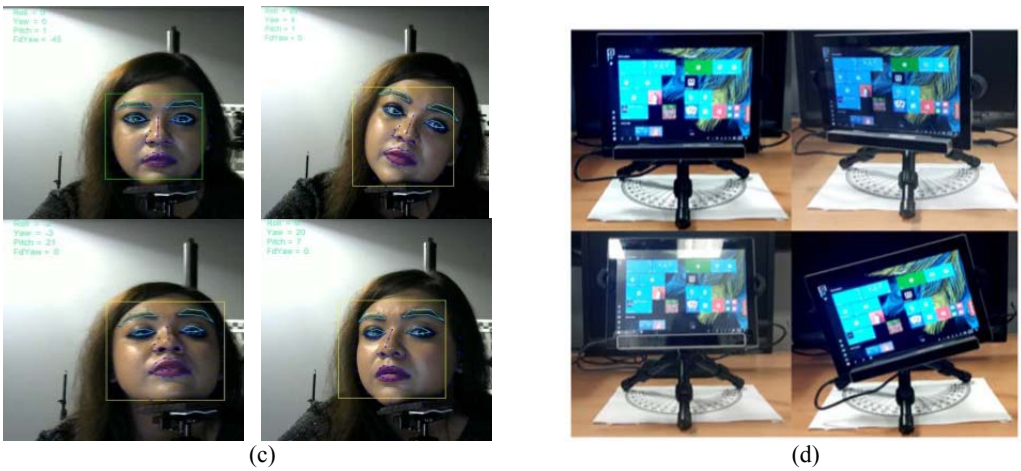
Eye tracking experiments were performed under different arrangements of the user and/or the eye tracker setup, which are termed as "operating conditions" in this thesis. The gaze data collection experiments done under different user-tracker configurations are described below.

(a) User distance experiments: In these, gaze data was collected at user-eye tracker distances of 50, 60, 70 and 80 cm. These experiments were designed to study variability of gaze tracking accuracy with visual angle which is inversely related to user-tracker distance. These experiments were performed on both the desktop and tablet setup.

(b) Head-pose variability experiments: This is relevant to studying the effect of a user's head pose on gaze tracking accuracy of the eye tracker. By head pose, the position of a user's head in 3D space in terms of roll, pitch and yaw (RPY) angles is meant. These experiments were done with the desktop setup. A user was seated at a fixed distance (60 cm) from the eye tracker and was asked to vary their head position to different rotation angles (head pose in roll, pitch, yaw and neutral) while looking at the UI on the display and their gaze locations on the UI were tracked (Figure 4.2c). The head position is also tracked simultaneously using a head pose model that measures head pose angles in RPY values with 1 degree of accuracy. Gaze data were recorded for each participant as follows:

- Head roll (R) plus (10, 20, 30 degree), roll (R) minus (10, 20, 30 degree)
- Head pitch (P) plus (10, 20 degree), pitch (P) minus (10, 20 degree)
- Head yaw (Y) plus (10, 20, 30 degree), yaw (Y) minus (10, 20, 30 degree)
- Neutral: The neutral pose is the frontal head pose where RPY= 0.

The various head poses are shown in Figure 4.2 c.



(c) (d)

Figure 4.2: (c) Face model showing different head poses (d) different platform poses of tablet with the eye tracker mounted.

(c) Platform orientation experiments: To quantize the impact of the orientation of an eye tracking setup on gaze data, experiments were performed in which the orientation of the tablet device (mounted with the eye tracker) was varied with respect to the user at fixed platform roll, pitch and

yaw angles. The tracker-tablet setup was placed on a flexible tripod mount such that the setup orientation could be varied as shown in Figure 4.2 d. Eye tracking data was collected for each tablet orientation with the same test UI as used for the desktop system. Gaze data in these experiments were collected for the following tablet-tracker poses:

- Tablet roll plus 20 degree, roll minus 20 degree
- Tablet pitch plus 10 degree, pitch plus 20 degree
- Tablet yaw plus 20 degree, yaw minus 20 degree
- Neutral where tablet roll, pitch, yaw = 0 degree.

In all the above experiments, for each variation of the setup or experiment condition, eye tracker calibration was done and calibration quality was tested. This was done using the validation option provided in the Tobii eye tracker's software. This option provided a set of validation targets, where the quality of calibration could be tested. The validation test was done after every calibration process to ensure the tracked user gaze is within the target boundaries of the validation screen. If the deviation was large, the calibration process was re-run.

The data collected from each experimental session in this work comprises of a participant's gaze coordinates, left and right eye positions and corresponding time stamps as estimated by the eye tracker. The ground truth data comprises of the stimuli locations of the UI in screen pixel coordinates. All data are stored in comma separated values or CSV files. Each data file has a naming convention which helps to label the data according to the operating conditions under which it was collected. More details about the collected dataset and labelling are described in Chapter 5.

4.2 Development of metrics and visualizations for gaze data evaluation

Eye tracking accuracy is typically measured in terms of the difference between the real stimuli (or target) positions and corresponding measured gaze positions. Several metrics that express eye tracking accuracy in angular measures, provide information about an eye tracker's inherent characteristics and also impact of variable operating conditions are developed in this work. These metrics are built using raw gaze data collected from the experiments described above and aim at quantitatively describing the quality of eye gaze data and its variabilities due to different operating conditions. These are summarized in Section 4.2.1.

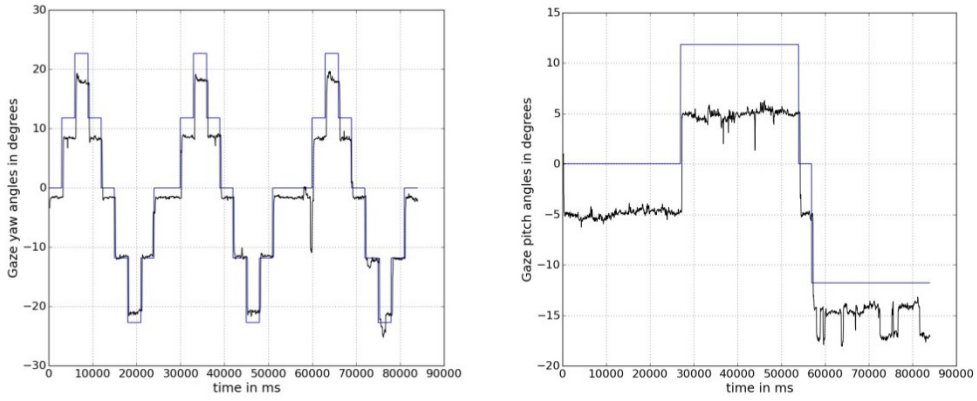
In addition to the gaze accuracy metrics, developing data visualization methods are crucial for aggregating large volumes of gaze data that are collected from eye tracking experiments, and for comparing gaze data characteristics like error magnitudes, biases and impact of variable operating conditions. Using the data collected from the eye tracking experiments described above, several graphical methods are developed in this work and described in Section 4.2.2. These visual tools may be used to explore the data characteristics of one or more eye trackers and study gaze estimation error

patterns when any operating condition is altered. Overall these visualizations are designed to aid in detailed characterization and comparison of gaze data quality from single or multiple eye trackers.
All these metrics and visualizations have been discussed in different sections of the paper [13] .

4.2.1 Summary of developed gaze data evaluation metrics

A. Gaze angular accuracy and yaw/ pitch accuracy metrics

A full derivation of gaze estimation accuracy (in degrees) and several gaze angular variables is made in Section 4.1.1 of the paper [13] which is in Appendix D of this thesis. Accuracy is estimated from the angular deviation between ground truth and tracked gaze location data from the experiments. The accuracy (or "gaze error") forms the fundamental variable for further analysis in this thesis.
The eye also undergoes rotational motion which leads to different eye orientations relative to the head. The two eye rotational variables are gaze yaw and pitch, where the yaw variation corresponds to left-right and pitch variation corresponds to top-bottom eye movements. The example errors corresponding to yaw and pitch movements of the eye are estimated and plotted in Figure 4.3 below.



(a) (b)

Figure 4.3: (a) Gaze yaw angle variations and (b) pitch angle variations overlaid on respective ground truth. The data is from one single experimental session for one person.

B. Statistical Metrics

A range of gaze error statistical measures are presented in Section 2 of the paper [13] (also in Appendix D) to analyse gaze errors from data collected for a group of individuals as done in this work. In practice, gaze experiments are performed on a group of subjects with group sizes ranging from 5–30 participants for improving test reliability. For analysing gaze data from numerous subjects, various statistical parameters must be evaluated to gain insight into gaze error patterns. In this work, apart from mean and standard deviation, statistical measures like 95% confidence interval and Z-score of gaze accuracy are shown to be significant indicators of gaze data quality. A large confidence interval indicates higher data variability and vice versa. The Z-score indicates the level of unusual data points or outliers within the dataset [129], and a high Z-score indicates noisy and scattered data.

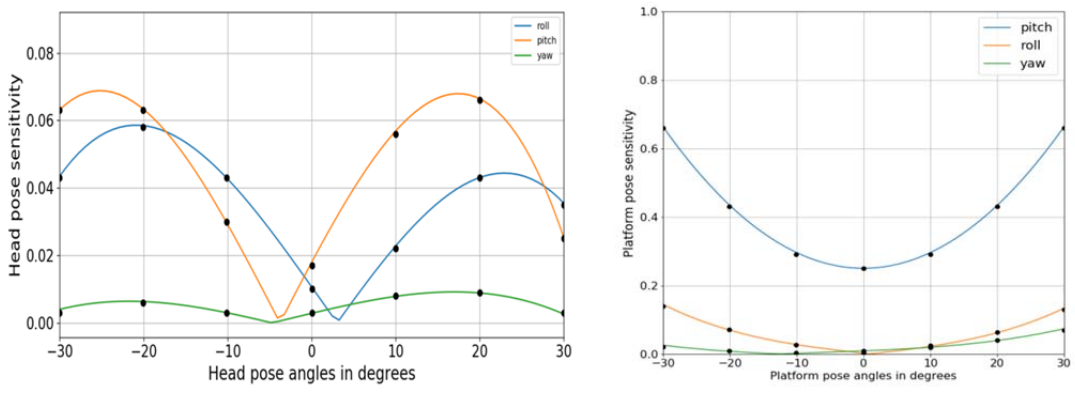
Histogram based metrics are proposed as additional statistical methods for evaluating and comparing gaze data[130] [131]. If H1 and H2 are the histograms of gaze errors from two gaze datasets, then correlation and intersection measures and Bhattacharya distance measure may be used to observe the similarity or divergence between the two datasets. This metric may be used to compare data from either two eye trackers, or between collected gaze data and reference gaze datasets.

Example results from using three different histogram based comparison techniques have been presented in Section 4.2.2 of the paper [13] (in Appendix D) where similarity between gaze datasets collected from different user distance experiments are analysed using this metric.

C. Sensitivity metrics

Data from our experiments were used to define and test two gaze-error-sensitivity metrics (representing head pose and platform pose sensitivity). The derivations of these metrics are described in details in Section 4.3.1 and 4.3.2 of [13] (in Appendix D). The concept and utilities of these two metrics are summarized below.

The head pose sensitivity metric is defined to represent how the variations of a user's head pose may affect the accuracy of an eye tracker under test. The manufacturers or designers building eye trackers do not quantitatively specify how much the tracking accuracy may deteriorate if user head pose is varied. For deriving this metric, data from our head-pose variability experiments is used. The head pose sensitivity curves for the eye tracker used in our work are shown in Figure 4.4 (a). It is seen that lowest errors occur for frontal head positions and errors increase rapidly as magnitude of head pose angles increase. Overall this metric shows that the head pose tolerance of our used eye tracker is quite low, and to achieve reliable gaze tracking, a user head movements must be constrained.



(a) (b)

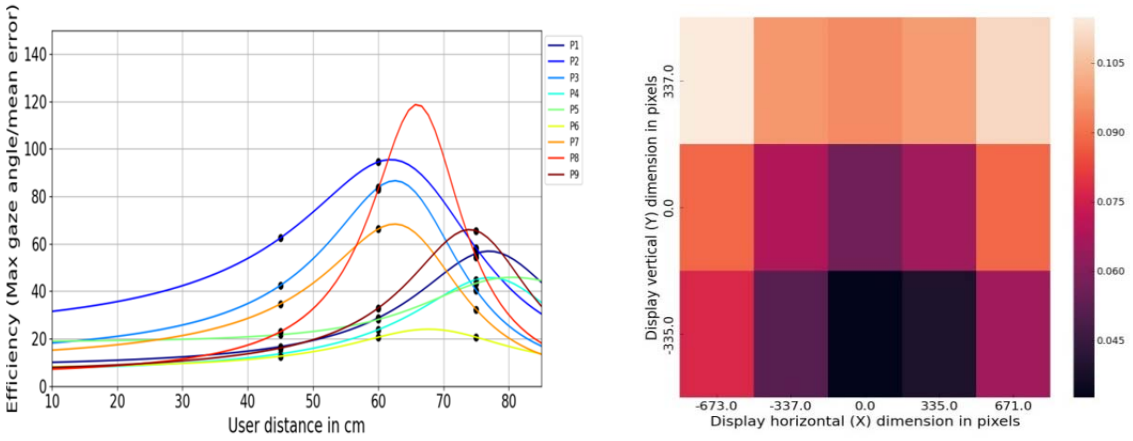
Figure 4.4: (a) Variation of head pose sensitivity as a function of head pose angles. (b) the gaze error sensitivity to orientations of the tablet. Black markers in the plots show the head and platform poses for which gaze data were collected.

To study the impact of platform orientations on the accuracy of an eye tracker, the platform pose sensitivity metric is defined (detailed derivation of this metric may be found in Section 4.3.2 of [13]

or in Appendix D). This metric is useful for eye tracking and gaze applications on handheld devices like smartphones and tablets, which face reliability issues due to the highly dynamic nature of the devices. The metric is implemented on the data collected from our tablet pose experiments and the plot of platform orientation sensitivity as a function of platform pose angles is shown in Figure 4.4b.

D. Gaze tracking efficiency metric

This metric is relevant to studying the impact of user distance and viewing angle on gaze estimation error. For implementing this metric, data from our user distance experiments done with the desktop setup is used. The gaze tracking efficiency as a function of user distance is plotted in Figure 4.5(a) below. Mathematical derivation of this metric and its background concepts may be found in Section 4.3.3 of [13] or in Appendix D). In a remote eye tracking setup, it is not known where the users should be ideally positioned in front of the eye tracker to achieve best results. The gaze tracking efficiency metric may be useful to quantitatively estimate for which user-tracker distance the best tracking accuracy can be obtained for a given tracker. Figure 4.5 (a) shows that gaze tracking efficiency increases as the user distance from the eye tracker increases until a certain value, after which gaze errors show an increasing trend. The best gaze tracking performance is achieved between user distances of 65–70 cm for the tracker used in our study.



(a) (b)

Figure 4.5: (a) Gaze tracking efficiency vs user distance (black markers are points where gaze data were collected, i.e. 45, 60, 75 cm) (b) spatial gaze error heat map for the display screen where gaze was tracked during the experiments.

E. Error Spatial Density metric

Gaze error magnitudes or statistics do not provide estimates about the spatial dependence of gaze error values on the screen where a user's gaze is tracked. For this purpose, a gaze spatial error density metric is defined to represent the area wise distribution of gaze errors on the screen. The metric is implemented using gaze data from our experiments and presented as an error heatmap shown in Figure 4.5 (b). Detailed derivation of this metric, including calculation of error spatial density from collected gaze data may be found in Section 4.3.4 of [13] or in Appendix D).

This metric can help to identify whether the gaze tracking accuracy is uniform over the monitor display and detect most probable locations on the screen where errors are large. This may help to improve gaze tracking performance, for example by checking the display screen quality, eye tracker mounting issues or compensating for errors if non-uniform error densities are observed.

F. ROC Metric for subjective performance evaluation of eye tracking systems

For in-depth performance analysis of an eye tracker using its data, the concept of a subjective performance evaluation metric is introduced in this work. Objective or absolute performance measures like angular accuracy are independent of any criteria set by an observer while subjective performance measures may depend on specific accuracy thresholds set by the observer. Benefit of using a subjective evaluation measure like the ROC metric developed in this work is that with these metric, users can set and observe the effect of varying gaze error thresholds and determine how to obtain best performance from a given tracker. They may also know if a certain tracker is suitable for them according to the required error threshold for a specific application.

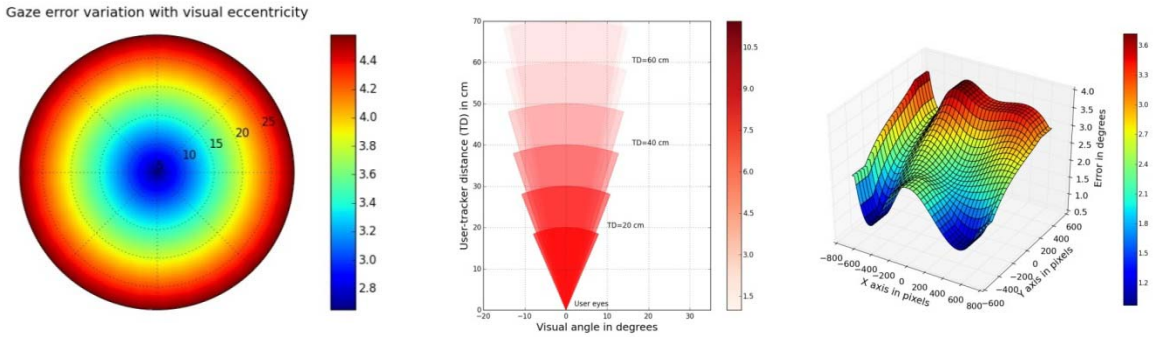
This metric is presented in details in Section 4.3.5 of [13] or in Appendix D, where the mathematical background for deriving this metric is established and example data from user distance experiments is used for establishing the functioning and utility of this new subjective metric for gaze data evaluation. Also, illustrative examples are provided to discuss how this new metric may be used by researchers or system designers for judging whether a given tracker is suitable for their application.

4.2.2 Summary of visualizations developed for exploring gaze data

A. Plot of gaze error as a function of visual eccentricity

The locations of gaze targets on the display screen of an eye tracking setup may have a significant effect on the quality of gaze data obtained from an eye tracker. A given tracker may show good accuracy in lower visual eccentricity areas (near the center of the display) and worse performance in high eccentricity areas (towards corners of the screen). Therefore, measuring gaze error characteristics with respect to stimulus eccentricity is valuable in specifying the level of gaze errors under various operating conditions for a given tracker.

The concept of using visual eccentricity to map gaze errors is presented in Section 4.4 of [13] or in the paper of Appendix D. Figure 4.6a below shows the corresponding visualization, which is polar plot where gaze error levels are plotted with respect to visual eccentricity values (varying between 5 to 30 degrees) on the display screen, using data from user distance experiment. The gaze errors at the center are found to be minimum and they increase with higher visual eccentricities near the corners.



(a) (b) (c)

Figure 4.6: (a) Polar plot of gaze error (degrees) mapped with respect to visual eccentricity (degrees) (b) visual angles of a user (degrees) color-mapped according to gaze error (degrees). (c) 3D plot of gaze errors (degrees) s a function of display X, Y dimensions (pixels)

B. Plot of gaze error vs. visual field

A plot of the dependence of gaze estimation accuracy on user visual angle (red sectors) and user distance from an eye tracker was described in Section 5.1.3 of [13] (in Appendix D) and shown in Figure 4.6b, which is implemented using data from our user distance experiments on desktop. The plot shows the reduction in user viewing angle with increasing user distance from tracker (shown by narrowing sectors). Also it shows that the eye tracker achieves best accuracy at a narrow visual angle of less than 20 degrees. These plots, along with plots like Figure 4.6a can help to know where to position a user in front of an eye tracker for best tracking results.

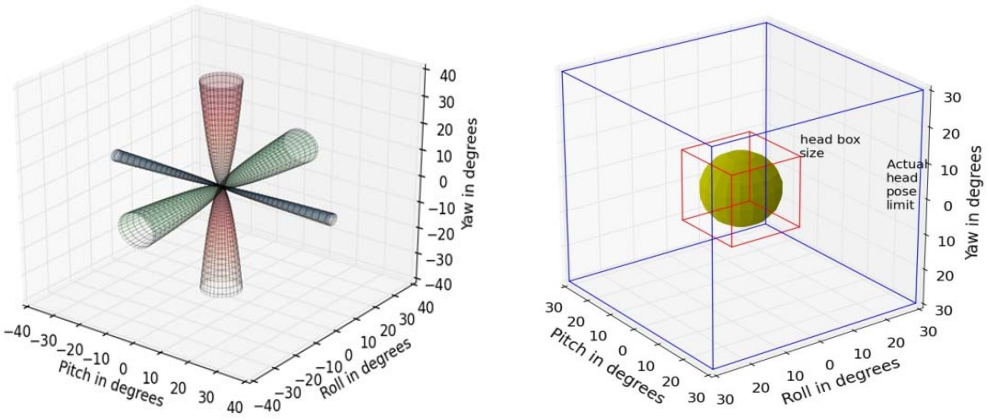
C. 3D gaze error distribution plot

A 3D gaze error distribution plot is constructed using gaze data from a user distance experiment and is shown in Figure 4.6c which is also presented in Section 5.1.4 of [13] (in Appendix D). It shows the magnitude of gaze errors as a function of X and Y dimensions (in pixels) of the viewing area on the display screen, while the gaze errors are plotted along Z axis. These plots help to diagnose gaze error levels over the display area. For example Figure 4.6c shows that high error values occur near the display corners and near the screen borders which could be due to high visual angles in those regions.

D. Tolerance plots

The concept of head pose and platform movement tolerance conics are discussed and implemented in Section 5.1.5 and 5.1.5 of [13] (Appendix D). These are shown below in Figure 4.7a and 4.7b respectively. The platform movement tolerance conics represent the degrees of freedom of movement "allowed" in a handheld device about its central axis if eye tracking is to be done on it with sufficient accuracy. Gaze error values obtained from platform orientation experiments are used for this plot. The utility of this visualization is that it shows the practical movement limits and pose variations that are allowed for a certain tablet-tracker setup to perform reliable eye tracking on it.

The head pose tolerance plot of Figure 4.7 b is implemented using data from head pose variability experiments. In the plot, the larger box shows the maximum degree of head movement (in roll-pitch-yaw angles) possible by an average user. The plot also shows that for reliable gaze estimation with the given tracker, the head pose of a user must be limited within the smaller "box", i.e. within 10 degrees in each angular direction to maintain gaze error magnitudes of 0.5 degrees. Such plots help to study the head-movement tolerances of eye trackers quantitatively.

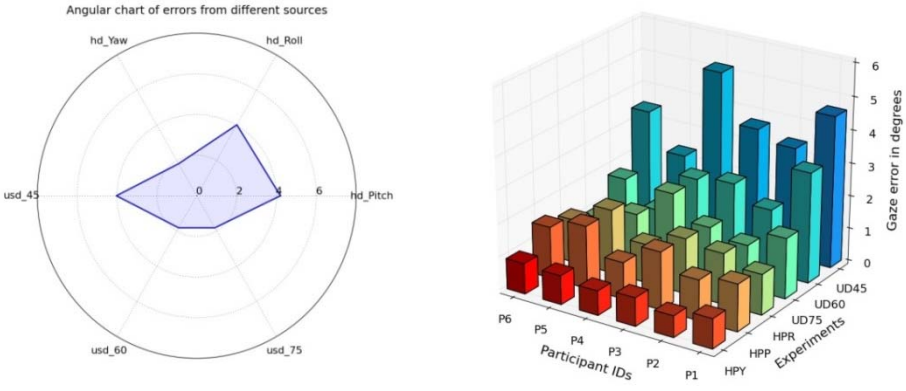


(a) (b)

Figure 4.7: (a) Visualization showing platform orientation tolerance of an eye tracker mounted on a tablet. The conics represent the maximum pose angles to which the tablet can be oriented while maintaining 1 degree of eye tracking accuracy. It is seen that platform pitch variations are more limited compared to yaw or roll variations, as derived using data from our platform pose experiments. (b) Head pose tolerance of an eye tracker shown graphically. The small bounding box represents the maximum head poses allowed for the given eye tracker while maintaining 0.5 degree of gaze error. The larger box represents head pose angles allowed under free head movement.

E. Data aggregation plots for multiple eye tracking experiments

In order to visualize data from more than one eye tracking experiment and when there are more than one factor to measure and compare while reporting eye tracker performance, angular charts as of Figure 4.8a could be useful. This chart is implemented using data from user distance (desktop) and head pose experiments and described in more details in Section 5.2.3 of [13] (Appendix D).



(a) (b)

Figure 4.8: Visualizations for gaze data aggregation. (a) An angular chart that shows the relative magnitudes of gaze error from different operating conditions (b) A very compact representation of gaze error data from all desktop experiments for 6 participants clustered into a single plot. These plots could be useful for comparing multiple tracking systems based on multiple criteria and help identify the better ones.

Another type of plot (Figure 4.8b) is implemented in Section 5.2.4 of [13] (Appendix D) to aggregate data of multiple persons from our user distance and head pose experiments to study and compare the gaze error levels of individuals across different experimental conditions. In the plot, different experiments are coded by colours with respective error values represented by bar heights, plotted along Z axis. This kind of plot can help inspecting data from several participants and judge overall eye tracking performance and draw quick and meaningful inferences about multiple experiments.

4.3 Software tools developed for performance evaluation of eye trackers

A new software tool, named *GazeVisual is* developed as part of this thesis work, specifically for the evaluation of an eye tracker's data characteristics and to quantitatively estimate and visualize the impacts of practical operating conditions on its performance. The software is in the form of a graphical user interface or GUI application which has a suite of statistical and visualization functions for in-depth analysis of eye tracker data. GazeVisual is built using Python programming language and Python libraries, and its main components are Python functions for 1) creation of the GUI application interface 2) incorporation of the GUI elements in the interface to input gaze data in CSV formats 3) implementation of methods from various Python data analysis and graphics libraries on the input gaze data to produce numerical measures and statistics of gaze accuracy levels and visualizations of gaze data characteristics. 4) possible generation of visual stimuli which can be displayed on screen to collect sample data from an eye tracker and also possibilities for direct interfacing of the software with an eye tracker for direct data collection. Details of the concept of this software, motivation for building it and description of its components and functionalities may be found in [63] and in the paper of Appendix E of this thesis. The source code of the GazeVisual software is also provided in an open repository on GitHub for eye gaze researchers or engineers to upgrade and extend (more details of this open repository are in Chapter 5). The link to the repository is:
https://github.com/anuradhakar49/GazeVisual-Lib/tree/master/Code%20repository/GazeVisual%20GUI%20tool

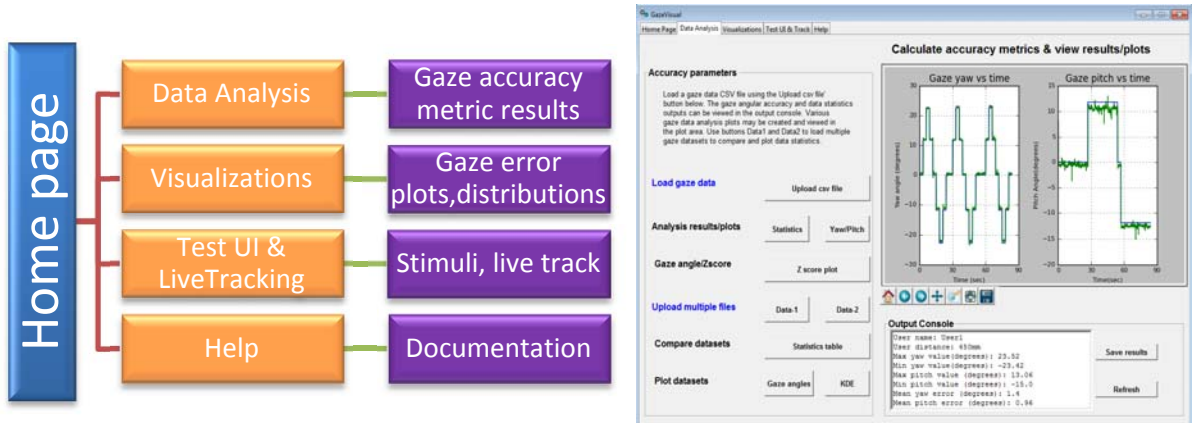
4.3.1 The GazeVisual software application tool

The GazeVisual software is intended to be a unified platform for quantitative and standardized evaluation of eye trackers using only their data. The gaze data format accepted by the software is described in more details in Chapter 5 and also in the paper of Appendix E. It has been tested with various samples of gaze and ground truth data from our experiments to produce numerical and graphical evaluation results and comparison of multiple eye gaze datasets. It has also been interfaced with an Eyetribe eye tracker to collect sample gaze data through its API (Figure 4.9d).These aspects are discussed in more details in Sections IV and V of the paper of Appendix E.

The architecture of the GazeVisual software is shown in Figure 4.9a below. The software has four separate windows. The major analytical and graphical functions of the software tool may be found in

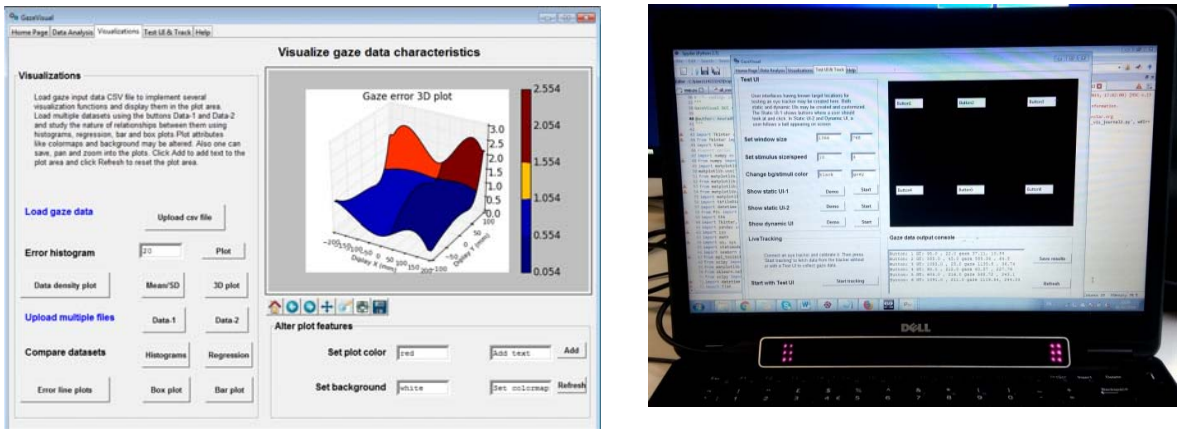
the “Data analysis” and “Visualizations” windows. The Test UI window contains functions for generating static or dynamic stimuli, connecting to an eye tracker and saving of the collected gaze data. Each window works independently and is designed to be easy to navigate and use by generic users. Plots generated from the “Data analysis” and “Visualizations” windows may be saved in PNG file formats. Evaluation results printed in the output consoles may be saved in text files.

Detailed description of the components of the “Data analysis” and “Visualizations” windows of GazeVisual software may be found in subsections A and B of Section IV of the paper in Appendix E. Details about the TestUI & LiveTracking concept and implementation are in Section 4C. Evaluations of the GazeVisual software using data from three different eye trackers are in Section V of this paper. Demo videos of the software have been created to show how the above different windows of this software may be used. Links to the videos may be found in the following GitHub repository. https://github.com/anuradhakar49/GazeVisual-Lib/blob/master/Code%20repository/GazeVisual%20GUI%20tool/Sample%20videos



(a) (b)

Figure 4.9: (a) Four windows of GazeVisual GUI tool and their functions (b) View of the Data Analysis window.



(c) (d)

Figure 4.9: (c) Visualizations window of GazeVisual (d) Test UI window of GazeVisual interfaced with an eye tracker

4.3.2 The GazeVisualApp web-application tool

In addition to the software tool, a web application named GazeVisualApp is developed in this work to implement a basic set of functions present in the GazeVisual GUI software over a cloud environment. This has several advantages, e.g. possibility of eye tracker evaluation on both desktop and mobile operating systems[132], accessing the evaluation functions independent of Python installations, ubiquitous accessibility and continuous updates of evaluation functions.

GazeVisualApp has similar analysis and visualization capabilities as the GazeVisual desktop GUI application and accepts raw gaze data in CSV format to provide corresponding gaze data quality metrics and visualizations through a web-browser. The web interface contains a single file upload button bar (Figure 4.10a). A user only needs to browse and upload a single CSV file having the gaze and ground truth data via this file upload button. The contents of the CSV file are used to estimate gaze data accuracy metrics and visualizations as done in the GazeVisual GUI tool. Input format of the CSV file for the web-application is similar as the input format for the GazeVisual GUI software.

After uploading the CSV file, the file contents are displayed on the browser window along with several data statistical measures (Figure 10a)) and six plots are created below the file upload bar (Figure 4.10b). All functions of the web-application take place on a single web browser page and all the plots may be saved in PNG format.
More details about the web-application design and implementation may be found in Section VI of the paper in Appendix E. The web application is hosted on a web server and currently live for testing at the address: http://gazevisual.pythonanywhere.com

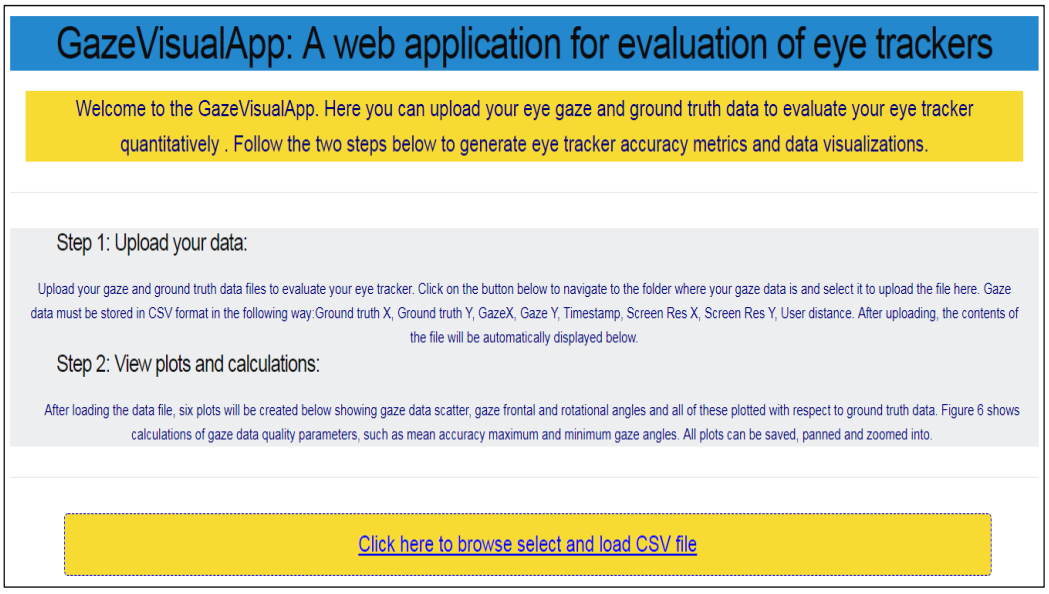
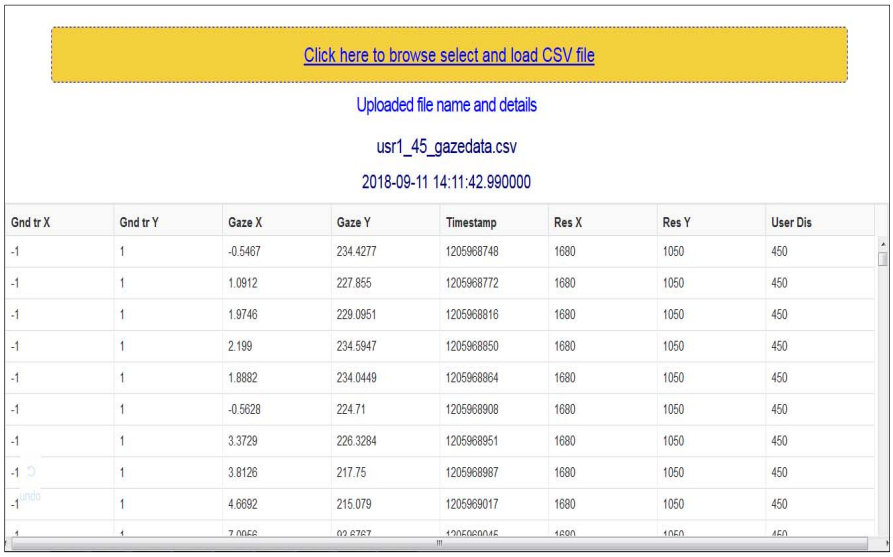


Figure 4.10: (a) View of the GazeVisualApp web application



Gnd tr X	Gnd tr Y	Gaze X	Gaze Y	Timestamp	Res X	Res Y	User Dis
-1	1	-0.5467	234.4277	1205968748	1680	1050	450
-1	1	1.0912	227.855	1205968772	1680	1050	450
-1	1	1.9746	229.0951	1205968816	1680	1050	450
-1	1	2.199	234.5947	1205968850	1680	1050	450
-1	1	1.8882	234.0449	1205968864	1680	1050	450
-1	1	-0.5628	224.71	1205968908	1680	1050	450
-1	1	3.3729	226.3284	1205968951	1680	1050	450
-1	1	3.8126	217.75	1205968987	1680	1050	450
-1	1	4.6692	215.079	1205969017	1680	1050	450

Figure 4.10: (b) Output plots and data statistics obtained from the GazeVisualApp

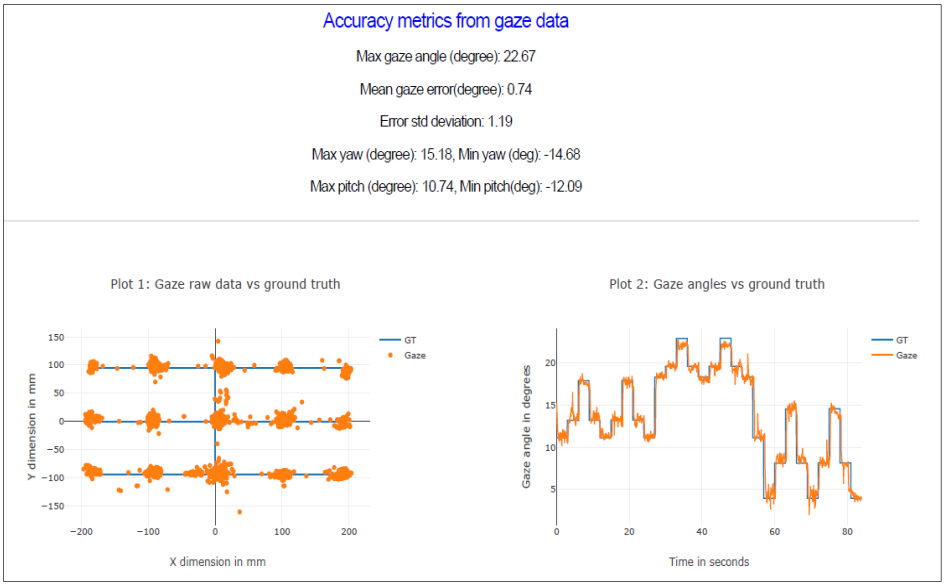


Figure 4.10: (c) GazeVisualApp displaying data in tabular form after uploading

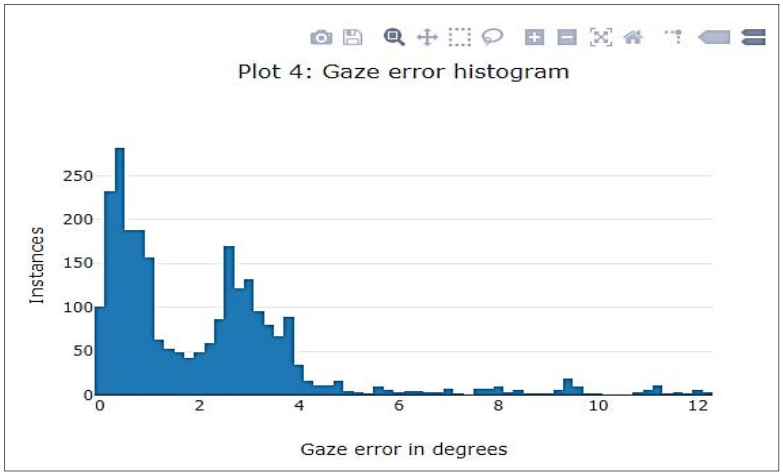


Figure 4.10: (d) A sample gaze error histogram plot obtained from the GazeVisualApp after uploading a gaze data file.

4.4 Machine learning models for analysing error patterns in gaze data

As discussed in this and the previous chapters, gaze data obtained from eye trackers operating on various consumer platforms is frequently affected by a multitude of factors, also termed as error sources in this thesis. As discussed in this chapter in subsections 4.1 to 4.3, a variety of methods and tools were developed as part of this thesis to quantitatively evaluate gaze data quality and study the impact of variable operating conditions on it. However, numerical metrics and visualizations often do not give adequate idea how gaze error patterns are caused due to different operating conditions. This is another aspect which is very sparsely explored in eye gaze research, i.e., identification and prediction of gaze error patterns caused by different operating conditions or error sources. In addition, there are currently no eye gaze datasets which have been collected under the influence of these error sources and labelled correspondingly, so that gaze error patterns induced by each distinct operating condition may be classified.

Based on the above observations, in this thesis work, statistical and machine learning algorithms are implemented to identify error patterns in eye gaze data, classify gaze error types and predict errors produced by an eye tracker when they operate under practical unconstrained scenarios. For this, the gaze datasets acquired from the desktop and tablet platform (as discussed in Section 4.1) were used. These datasets were labelled with the different conditions (user distances, head pose, platform pose) and classifiers were trained on them to identify the influence of these conditions. Also, regression models were built to predict the variability in gaze error levels under the influence of these conditions.

This section presents a summary of the methods which were developed to pre-process raw eye gaze data and successively detect and predict gaze error patterns caused by multiple operating conditions using classifier and regression models. These methods are aimed to provide in-depth knowledge above an eye tracker's gaze data characteristics and improve the quality and reliability of eye trackers operating under unconstrained operating conditions.

Full details about the background concepts on gaze error pattern recognition, preparation of gaze datasets for the learning algorithms and results from the algorithms may be found in the paper which is in Appendix F of this thesis.

4.4.1 Research questions on gaze error pattern recognition

With respect to studying gaze error patterns, the following research questions were identified which have not been previously addressed in gaze research. 1) Do the different error sources produce any particular pattern of gaze errors or if the nature of gaze errors follows any statistical distribution or are they simply random. 2) How can suitable features be extracted from gaze error data to identify the error sources 3) how can gaze errors caused by one error source be distinguished from those caused

by another 4) how can the presence of different error sources in a gaze dataset be detected without prior knowledge 5) is it possible to predict the level of gaze errors caused by different error sources.

4.4.2 Steps for implementing machine learning for gaze error pattern analysis

With raw gaze data, it is impossible to see any existing error patterns in the data, since data from very different operating conditions often look similar and vice versa. Further, raw data is always corrupted with varied levels of outliers. Raw gaze data collected from our different experiments cannot be input directly to machine learning models for classification or prediction. Therefore, the data processing pipeline shown in Figure 4.11 is implemented. More details on the data preparation and exploration steps are discussed in Section 2.4 and 2.5 of the paper in Appendix F of this thesis.

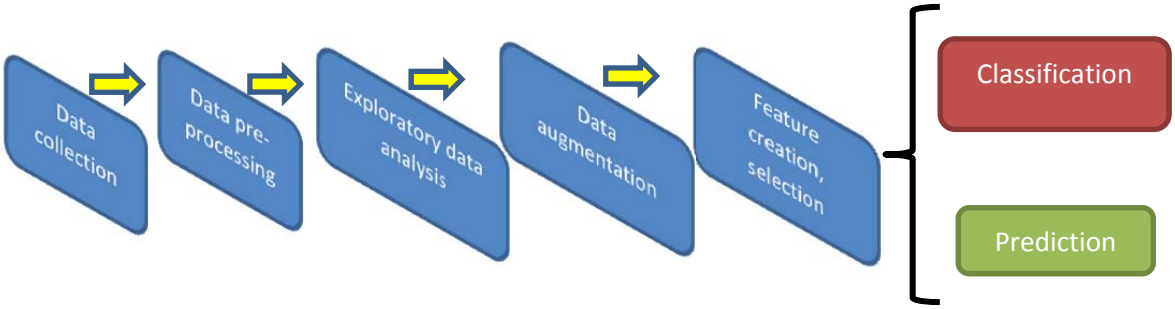


Figure 4.11: Data processing pipeline for gaze error analysis using machine learning

Data collected from the user distance and head pose experiments from desktop and user distance and platform pose experiments from tablet were used for the pattern recognition tasks. Firstly, outliers were removed using median filtering [133], followed by feature extraction from the data [134]. The following feature set is computed for all datasets:

$$[Error\_AOI\text{-}1,\ Error\_AOI\text{-}2 \ldots Error\_AOI\text{-}15,\ \mu,\ \sigma,\ IQR,\ 95\%\ conf\ upper,\ 95\%\ conf\ lower]_{sample}$$

Where Error_ AOI-1 to Error_ AOI-15 are the mean gaze errors at the 15 AOI locations (shown in Figure 4.1) and the other values are the statistical measures computed on each data sample. After feature estimation, samples from desktop datasets corresponding to 4 user distance and 3 head pose classes are mixed, and similarly data from 4 tablet distance and 3 platform pose classes are mixed. Details about feature extraction from collected gaze datasets is discussed in Section 3.3 of the paper in Appendix F. A comparison of gaze data from the desktop and tablet platforms under different head poses, user distances and platform poses are in Appendix K.

The following classification tasks were implemented on the desktop datasets: 1) classification of errors patterns for different user distances (i.e. between 4 user distance classes) (2) classification of head pose error patterns (i.e. between 4 head pose classes of neutral, roll, pitch, yaw), (3)classification between head pose and user distance errors patterns (7 class classification).

The following classification tasks were implemented on the tablet datasets: 1) classification of error patterns for different user distances (i.e. between 4 classes) (2) classification of tablet pose error patterns (i.e. between 4 tablet pose classes: neutral, roll, pitch, yaw), (3) classification between tablet pose and user distance errors patterns (7 class classification).

The prediction task included using regression models[135] to map between the different gaze angular variables with gaze error values from both desktop and tablet datasets, defined as below:

[Gaze_Angle, Gaze_Yaw, Gaze_Pitch] → Gaze error

The regression models were trained and tested on datasets belonging to the same category (i.e, user distance, head pose, platform pose) and model fit qualities were estimated from RMSE values.

4.4.3 Classification and regression models for gaze error pattern analysis

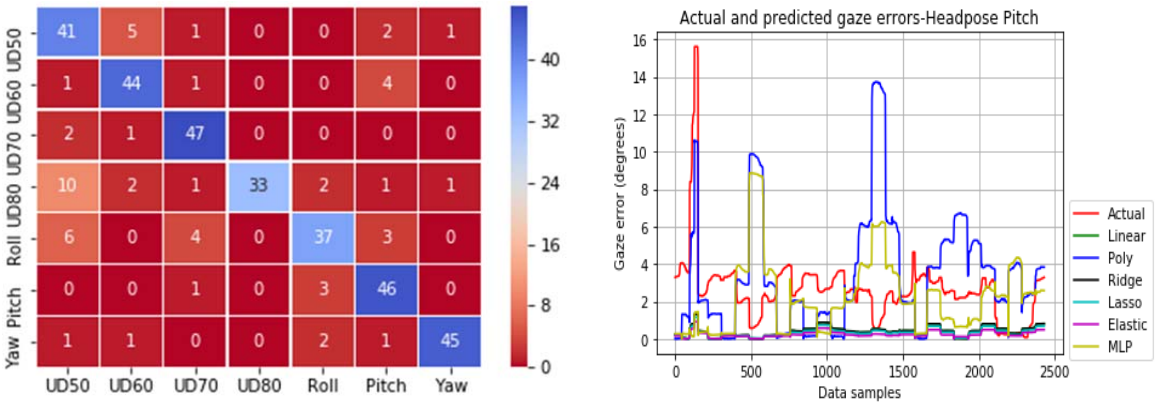
Three different machine learning models were used for the above classification tasks. These included k-Nearest Neighbours [136], SVMs [137] and Neural networks [135]. The models were optimized through experiments to select the set of hyper-parameters that best improve their cross validation scores. The classification problems where datasets from different categories were mixed were found to have complex feature distributions with little separation between classes. However, the machine learning models were still able to distinguish between features from the different datasets. Detailed discussions about these classifier models may be found in Section 3.4 of the paper in Appendix F.

For prediction of errors, six different regression models were used including linear, polynomial, Ridge, Lasso, ElasticNet and Neural network regressors [138] [139]. More details about these regression models are presented in Section 4 of the paper in Appendix F.

4.4.4 Outcomes of the learning models

The classifier models achieved on average a classification accuracy of 90% with 10 fold cross validation (Figure 4.12a) with the KNN and neural networks achieving close performance for all datasets. The classification results for the desktop and tablet datasets are presented and analysed in lot more details in Sections 3.5 and 3.6 of the paper in Appendix F, where aspects such as model hyper-parameter tuning, testing and cross-validation of model scores are discussed.

Out of the prediction models, ElasticNet showed the lowest RMSE of 0.7 for most of the prediction tasks (Figure 4.12b). All the models were seen to be highly sensitive to the presence of outliers and all the features therefore had to be standardized before input to the models. The results from the regression models are analysed in details in Section 4.1 (for models trained on desktop datasets) and Section 4.2 (for models trained on tablet datasets) of the paper in Appendix F.



(a) (b)

Figure 4.12: (a) Classification result from using Neural networks to distinguish between desktop user distance and head pose datasets (b) Output from a prediction task where head pose pitch error data was modelled using various regressors.

4.5 Conclusion

In this chapter, the implementation details of the proposed performance evaluation framework for remote eye tracking systems are summarized. The various concepts and components of the framework are presented in a condensed summary form and more detailed descriptions of the developed methods and software may be found in the referenced papers.

Researchers, engineers and designers working with gaze based systems can use the methods to find out the practical capabilities and limits of their eye trackers, perform in-depth assessment of eye gaze data collected from an eye tracker and understand under what circumstances their trackers can provide best results (or lowest gaze errors). The methods and tools, developed as part of this thesis work are given out as open resources for the eye gaze research community to use and upgrade. Details of the open repository containing resources for implementing these methods as well as the collected gaze datasets are discussed in Chapter 5.

Chapter 5

Open repository of resources for performance evaluation of eye trackers

In contemporary eye gaze research, there is a marked deficiency of open source tools, datasets and software which can be used for consistent evaluation of eye trackers and gaze data quality. Considering these requirements, several gaze data evaluation algorithms, visualizations and software tools which were developed as part of this thesis work are released into an open source repository on GitHub, and named GazeVisual-Lib. (https://github.com/anuradhakar49/GazeVisual-Lib/). This repository contains a range of resources implemented as Python codes that may be used to process raw gaze data collected from an eye tracker, estimate several gaze angular variables and accuracy in units of angular resolution and implement multiple evaluation metrics and visualizations. Further, the source code of the GazeVisual GUI software (described in Chapter 4) is also included in the repository. The GazeVisual-Lib repository is aimed to allow generic eye tracker users from a broad range of interdisciplinary areas to freely access different gaze data evaluation methods and also let researchers and advanced programmers working in these fields to modify the codes to suit their own application or upgrade them to more sophisticated forms.

In addition to the code repository, the datasets collected from our different eye tracking experiments as described in Chapters 3 and 4 are aggregated into an open data repository hosted on Mendeley Data named NUIG_EyeGaze01(Labelled eye gaze dataset). Link to the dataset is below.
https://data.mendeley.com/datasets/2txnw3y6gs/draft?a=b8ab5ca4-4d5e-463e-b5fc-a463d548102d
This is a new development in gaze research as the dataset is labelled with a wide range of operating conditions and could be of significant interest to gaze researchers and engineers for comparison and benchmarking of gaze tracking performance.
This chapter describes the concept and organization of the code and data repositories developed as part of this thesis, their components, data input and output formats and guidelines on how to use them. Also the utilities of the repositories towards gaze data analysis and evaluation are highlighted.

This chapter is based on one paper which is published [13] and two papers which are currently under review, and copies of which are in Appendices D, E, G.

5.1 A GitHub repository for gaze data evaluation methods

The GazeVisual-Lib repository provides a fully defined set of numerical and visual methods for in-depth evaluation and comparison of the data quality from generic/commercial gaze tracking systems. The methods are implemented as fully documented Python codes and sample data files and demo videos are also provided in the repository for using and testing the codes. The organization of the repository is shown in Figure 5.1, followed by description of its components and functionalities.

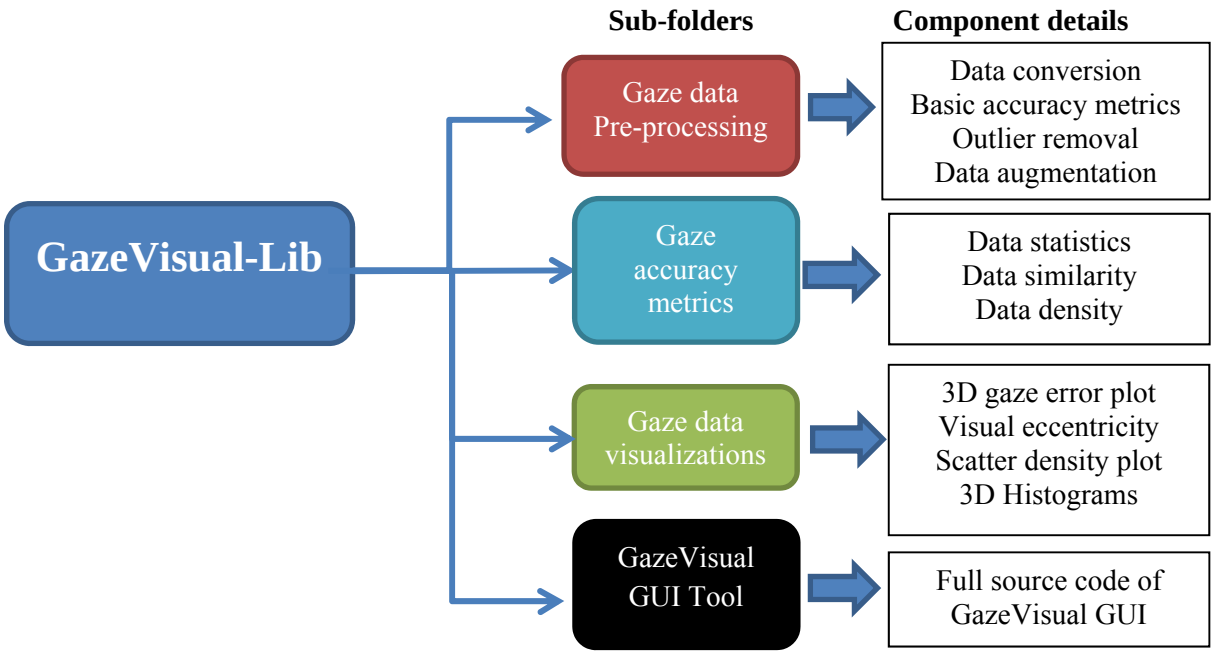


Figure 5.1: Organization of the GazeVisual-Lib code repository on GitHub

The GazeVisual-Lib repository is meant to provide easy to use gaze data evaluation resources in Python programming language, for free use, modification and upgradation by eye gaze researchers and engineers. The Python codes and supporting information for using the numerical methods and visualizations are included in the repository in different folders. The contents of these folders and their functionalities are provided below.

5.1.1 "Gaze data pre-processing" folder

The Python codes in this folder and their functions are in Section 2.2 A of the paper in Appendix G. There are three Python files which are used for data conversion and pre-processing. Data conversion refers to the estimation of gaze angular variables and accuracy from raw gaze data and corresponding ground truth data, as discussed in Section 4.1 of [13]. Data conversion methods are included in the main_proc.py file of this folder. The output of the data conversion process is a CSV (comma separated values) file which contains 16 columns of data derived from the raw gaze and ground truth

data. A sample output CSV file (named user_data_proc.csv) is also provided within this folder. The headers and contents of this output CSV file are shown below in Table 5.1.

Table 5.1: Description of column contents of "user_data_proc.csv" file

Header no.	Header title	Description of column values
1.	TIM REL	Relative time stamp for each gaze data point in the file
2.	"GTX"	Ground truth x positions in pixels
3.	"GTY"	Ground truth y positions in pixels
4.	"XRAW"	Raw gaze data x coordinates in pixels
5.	"YRAW"	Raw gaze data y coordinates in pixels
6.	"GT Xmm"	Ground truth x positions in mm
7.	"GT Ymm"	Ground truth y positions in mm
8.	"Xmm"	Gaze x positions in mm
9.	"Ymm"	Gaze y positions in mm
10.	"YAW GT"	Ground truth yaw angles from ground truth data
11.	"YAW DATA"	Estimated yaw angles from input gaze data
12.	"PITCH GT"	Ground truth pitch angles from ground truth data
13.	"PITCH DATA"	Estimated gaze angles from input gaze data
14.	"GAZE GT"	Ground truth gaze angles from ground truth data
15.	"GAZE ANG"	Estimated gaze angles from input gaze data
16.	"DIFF GZ"	Difference between ground truth and estimated gaze angles

From the data columns in the CSV file, gaze yaw and pitch angular accuracy values can be calculated by subtracting "YAW GT" from "YAW DATA" or "PITCH GT" from "PITCH DATA" columns. Input gaze x,y coordinates should have value 0,0 at the center of the display screen and the data should be free from invalid or "NAN" entries.

Another code in this folder is the "outlier_removal.py" file which implements three different outlier detection and removal strategies on collected gaze data. These are: 1D Median filtering, use of median absolute deviation and outlier detection using the interquartile range method [13].

The third code file in this folder is the data_augmentation.py which contains methods that may be used to expand an eye gaze dataset. These methods include addition of white and coloured noise, data interpolation, time-shifting, data convolution with cosine kernels and a combination of these[140]. Augmented or synthetic gaze datasets may be used for a variety of purposes such as use of machine learning algorithms[137] to model gaze data patterns of a tracker or errors caused by different factors.

5.1.2 "Gaze accuracy metrics" folder

The components of this folder are described in Section 2.2 B of the paper in Appendix G. In this folder, there are three python files (data_statistics.py, data_similarity.py, spatial_density.py) which may be used to compute gaze data statistics, similarity between gaze datasets and gaze error spatial density on the display screen where gaze was tracked. The data similarity calculations are based on correlation, intersection and Bhattacharya distance [13] computed on histograms of two gaze datasets. The scatter_density.py file helps to create a gaze data density plot using raw gaze data that allows users a quick look into the relative scatter of data points on the display screen during data collection, which makes it easier to interpret gaze data patterns and detect any anomaly.

5.1.3 "Gaze data visualizations" folder

The details of the visualization methods present in this folder are described in Section 2.2 C of the paper in Appendix G. In this folder there are three python codes that implement various visualizations as described in [13]. The file 3D_plot.py takes in the gaze error values (i.e. data from the "DIFF GZ" column of the user_data_proc.csv file) and creates a 3D distribution of gaze error magnitudes (plotted along Z axis) as a function of X and Y dimensions of the display screen. This plot helps to investigate and compare gaze error levels over the display area where gaze was tracked.

The eccentricity.py file creates a plot of gaze error levels mapped as a function of gaze yaw and pitch angle values on the display screen. This program may be used to study how gaze errors vary with visual angles, especially if user distance from the tracker is increased or decreased. Usually for shorter distances gaze errors are more sensitive to visual eccentricities, whereas gaze errors for long distances (e.g. at 80 cm) show less sensitivity to eccentricities [13].

The file 3D_histogram.py plots stacked data distributions using data from two or more trackers, persons or experiments in a single plot. It helps to understand and compare data patterns, and gain insight into data characteristics such as where error values are concentrated or presence of data extremes. The output of the code shows three gaze angular variables plotted together as stacked 3D histograms using a bin size of 20 (which can be varied).

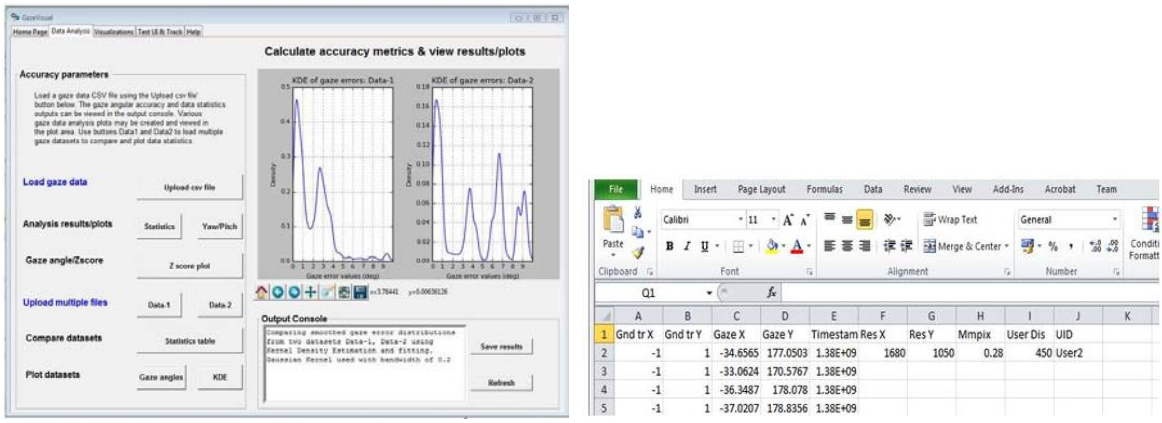
5.1.4 "GazeVisual GUI Tool" folder

This folder contains the source code for the GazeVisual GUI software described in Chapter 4. The purpose and functions of this software are discussed in detail in the paper attached in Appendix E.

The input data format for GazeVisual software is shown in Figure 5.2. It comprises of columns for raw gaze and ground truth data coordinates, display screen resolution and pixel pitch of the display (where gaze was tracked) and user distance from the tracker. Two sample gaze data files that can be

used as input to the GazeVisual software are provided in the folder. Links to demo videos showing how users may navigate through and implement different functions of the software are also included.

GazeVisual can run as a desktop application on any operating system having Python 2.7 and with Python libraries such as Tkinter, Pygame, Statsmodels and Seaborn installed. For interfacing an eye tracker with GazeVisual , it is required that the tracker's Software Development. Kit (SDK) and calibration routines are installed in the computer where GazeVisual runs. GazeVisual can also run alone using the gaze (and ground truth) data collected beforehand on the same or different computer.



(a) (b)

Figure 5.2: (a) View of the "Data Analysis" window of the GazeVisual GUI tool (b) input data format for the software

5.1.5 Cross-applicability of the evaluation methods for different eye-trackers

The gaze data evaluation methods in GazeVisual-Lib are based on calculations using data from both eyes and centralized gaze coordinates as provided by our eye tracker. If any eye tracker provides only monocular data or only the centralized gaze coordinates without the left/right eye position values, then the repository codes can still be used for gaze angle and accuracy calculations after minor changes. Thus, with monocular data (which are still gaze coordinates at the target points but using single eye data), the metrics will work but results may vary from the case when binocular tracker data is used.

The GazeVisual GUI application has been tested with data from two remote eye trackers and a head mounted eye tracker as described in Section V of the paper in Appendix E. The remote trackers and the head mounted tracker have different frame rates but both provide fixation data or estimated gaze coordinates. The GazeVisual software is seen to produce consistent results for both types of trackers with the requirements that: a) input gaze data comprises of fixation data and corresponding ground truth data b) input gaze and ground truth data coordinates have their origin at display center c) input data is arranged in the format shown in Figure 5.2b, d) the data is free from non-numeric or NAN values e) gaze and ground truth data have same lengths (number of data rows).

5.2 An open data repository for new benchmark eye gaze datasets

A rich and diverse gaze dataset, using the eye tracking data collected during our experiments under wide range of operating conditions (described in detail in Chapter 3 and 4) is built and presented in an open data repository as part of this thesis work. The dataset is named: NUIG_EyeGaze01(A labelled benchmark eye gaze dataset) and is hosted in the Mendeley open data repository. The organization of the dataset is shown in Figure 5.3. This is a new kind of dataset, collected from three user platforms (desktop, laptop, tablet) under the influence of one condition at a time, using a commercial remote eye tracker. The conditions include, 17 different head poses, 4 user distances, 6 platform poses and 3 display screen size and resolutions.

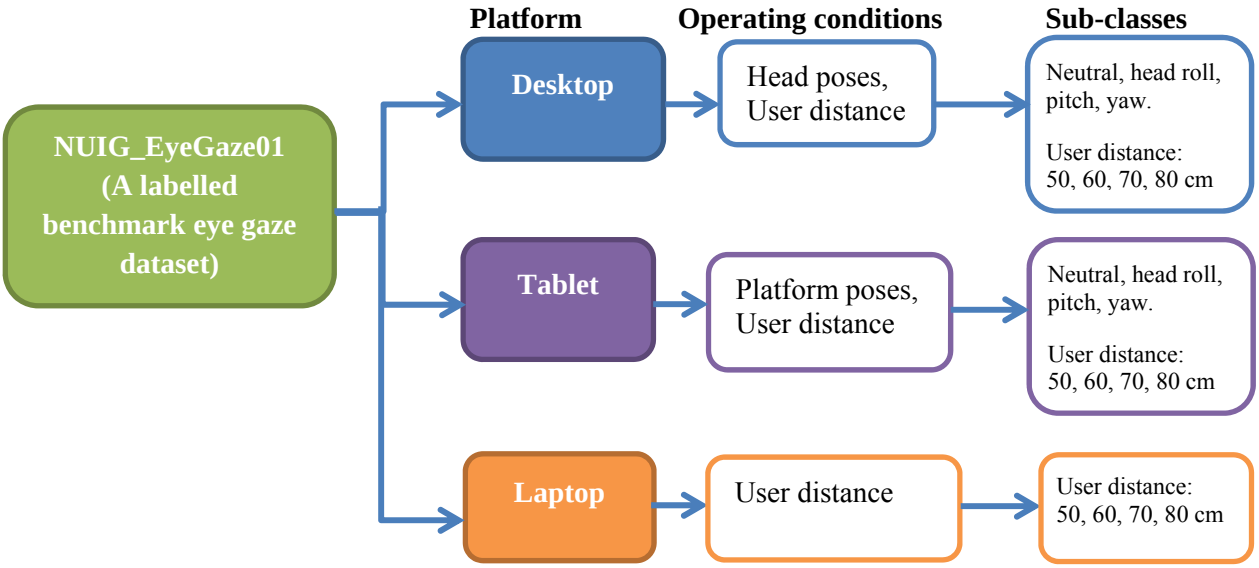


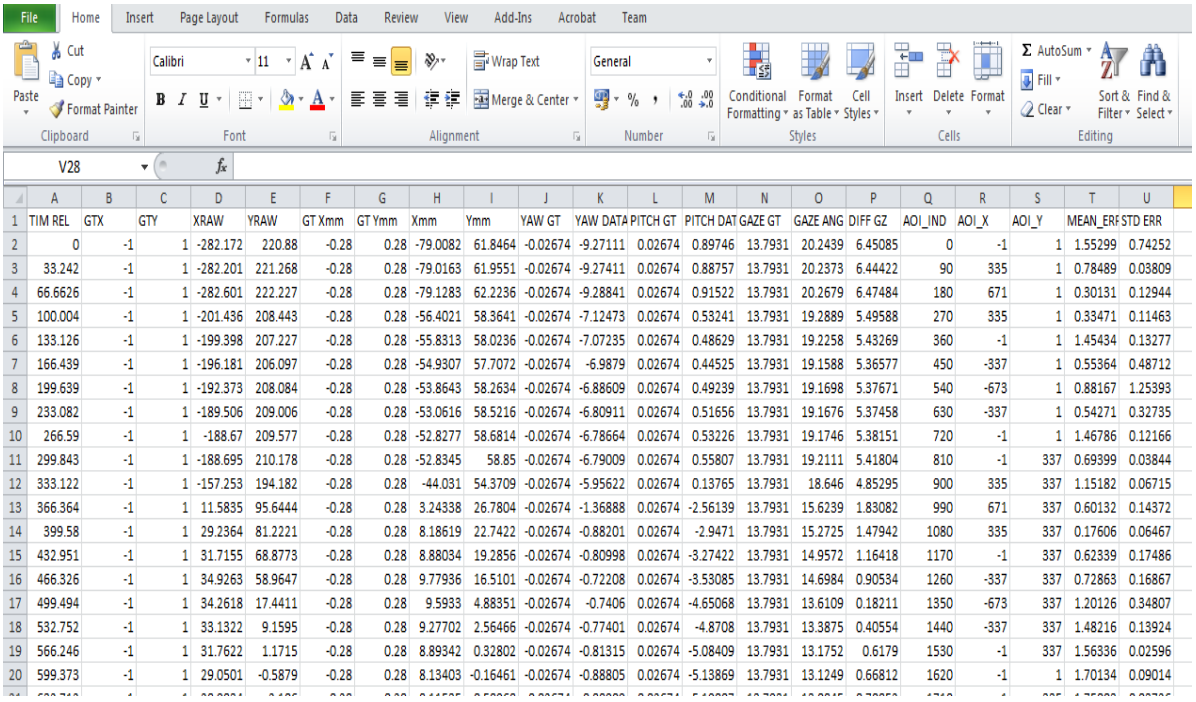
Figure 5.3: Dataset organization in the NUIG_EyeGaze01 repository on Mendeley data

Each gaze data file in the repository is labelled with the type of operating conditions under which it was collected. Name of each file has convention: USERNUMBER_CONDITION_PLATFORM.CSV (e.g. us01_80_desk.csv). Gaze data from 20 participants (15 male, 5 female) were collected for each condition mentioned above. Ground truth data for each platform is provided in the respective files, which are the actual locations of the stimuli dots appearing on display screen during gaze data collection. Each gaze data file contains 21 data columns (shown in Figure 5.4) corresponding to raw gaze data along with timestamps, gaze angular variables like gaze yaw, pitch values and gaze estimation accuracy for each gaze data point as well as for each stimuli location.

Details of the data collection process are in [13] and in Chapter 3 and 4 of the thesis. Other details regarding this repository are in Table 5.2; which are also provided in the main page of the online repository on Mendeley data. Link to this data repository is provided in the beginning of this chapter.

Table 5.2: Details of the NUIG_EyeGaze01 dataset capture conditions

Dataset parameter	Description
Data type	Fixations
Data file type	CSV
Collection device:	Tobii EyeX 4C tracker.Specified accuracy : 0.5 degrees (frontal head pose).
Collection platforms:	Desktop, tablet, laptop
Screen resolutions of platforms	22 inch (Desktop, 1680 x1050 pixels), 14 inch (Laptop, 1366x 768 pixels) 10.1 inch (Tablet, 1920 x 800 pixels)
Ambient illumination	Constant, 60 Lux
Others	Chin rest used for all sessions. No participants wore glasses.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	TIM REL	GTX	GTY	XRAW	YRAW	GT Xmm	GT Ymm	Xmm	Ymm	YAW GT	YAW DATA	PITCH GT	PITCH DAT	GAZE GT	GAZE ANG	DIFF GZ	AOI_IND	AOI_X	AOI_Y	MEAN_ERF	STD ERR
2	0	-1	1	-282.172	220.88	-0.28	0.28	-79.0082	61.8464	-0.02674	-9.27111	0.02674	0.89746	13.7931	20.2439	6.45085	0	-1	1	1.55299	0.74252
3	33.242	-1	1	-282.201	221.268	-0.28	0.28	-79.0163	61.9551	-0.02674	-9.27411	0.02674	0.88757	13.7931	20.2373	6.44422	90	335	1	0.78489	0.03809
4	66.6626	-1	1	-282.601	222.227	-0.28	0.28	-79.1283	62.2236	-0.02674	-9.28841	0.02674	0.91522	13.7931	20.2679	6.47484	180	671	1	0.30131	0.12944
5	100.004	-1	1	-201.436	208.443	-0.28	0.28	-56.4021	58.3641	-0.02674	-7.12473	0.02674	0.53241	13.7931	19.2889	5.49588	270	335	1	0.33471	0.11463
6	133.126	-1	1	-199.398	207.227	-0.28	0.28	-55.8313	58.0236	-0.02674	-7.07235	0.02674	0.48629	13.7931	19.2258	5.43269	360	-1	1	1.45434	0.13277
7	166.439	-1	1	-196.181	206.097	-0.28	0.28	-54.9307	57.7072	-0.02674	-6.9879	0.02674	0.44525	13.7931	19.1588	5.36577	450	-337	1	0.55364	0.48712
8	199.639	-1	1	-192.373	208.084	-0.28	0.28	-53.8643	58.2634	-0.02674	-6.88609	0.02674	0.49239	13.7931	19.1698	5.37671	540	-673	1	0.88167	1.25393
9	233.082	-1	1	-189.506	209.006	-0.28	0.28	-53.0616	58.5216	-0.02674	-6.80911	0.02674	0.51656	13.7931	19.1676	5.37458	630	-337	1	0.54271	0.32735
10	266.59	-1	1	-188.67	209.577	-0.28	0.28	-52.8277	58.6814	-0.02674	-6.78664	0.02674	0.53226	13.7931	19.1746	5.38151	720	-1	1	1.46786	0.12166
11	299.843	-1	1	-188.695	210.178	-0.28	0.28	-52.8345	58.85	-0.02674	-6.79009	0.02674	0.55807	13.7931	19.2111	5.41804	810	-1	337	0.69399	0.03844
12	333.122	-1	1	-157.253	194.182	-0.28	0.28	-44.031	54.3709	-0.02674	-5.95622	0.02674	0.13765	13.7931	18.646	4.85295	900	335	337	1.15182	0.06715
13	366.364	-1	1	11.5835	95.6444	-0.28	0.28	3.24338	26.7804	-0.02674	-1.36888	0.02674	-2.56139	13.7931	15.6239	1.83082	990	671	337	0.60132	0.14372
14	399.58	-1	1	29.2364	81.2221	-0.28	0.28	8.18619	22.7422	-0.02674	-0.88201	0.02674	-2.9471	13.7931	15.2725	1.47942	1080	335	337	0.17606	0.06467
15	432.951	-1	1	31.7155	68.8773	-0.28	0.28	8.88034	19.2856	-0.02674	-0.80998	0.02674	-3.27422	13.7931	14.9572	1.16418	1170	-1	337	0.62339	0.17486
16	466.326	-1	1	34.9263	58.9647	-0.28	0.28	9.77936	16.5101	-0.02674	-0.72208	0.02674	-3.53085	13.7931	14.6984	0.90534	1260	-337	337	0.72863	0.16867
17	499.494	-1	1	34.2618	17.4411	-0.28	0.28	9.5933	4.88351	-0.02674	-0.7406	0.02674	-4.65068	13.7931	13.6109	0.18211	1350	-673	337	1.20126	0.34807
18	532.752	-1	1	33.1322	9.1595	-0.28	0.28	9.27702	2.56466	-0.02674	-0.77401	0.02674	-4.8708	13.7931	13.3875	0.40554	1440	-337	337	1.48216	0.13924
19	566.246	-1	1	31.7622	1.1715	-0.28	0.28	8.89342	0.32802	-0.02674	-0.81315	0.02674	-5.08409	13.7931	13.1752	0.6179	1530	-1	337	1.56336	0.02596
20	599.373	-1	1	29.0501	-0.5879	-0.28	0.28	8.13403	-0.16461	-0.02674	-0.88805	0.02674	-5.13869	13.7931	13.1249	0.66812	1620	-1	1	1.70134	0.09014

Figure 5.4: A screenshot of the data format in each CSV file of the NUIG_EyeGaze01 dataset

Meaning of the data columns 1-16 are explained in Table 5.1above. The last 5 fields signify the locations of the stimuli dots and gaze error statistics measured at those points. User distance or other operating conditions during collection of data can be extracted from the respective file names.

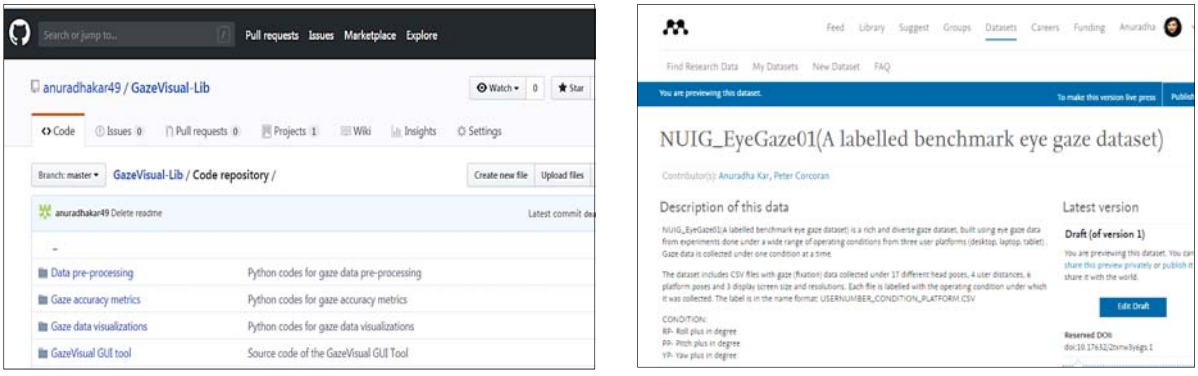
5.3 How to use the repositories

To run the codes in the GazeVisual-lib code repository, users must have Python 2.7 and the different libraries (mentioned in Section 5.1) in their latest versions installed in their computers. All the codes are in the "Code repository" folder (Figure 5.5 below) which has several "README" files in various sub-folders that provide details about how to format an input data file and use it with the codes. The README file of the root folder is the main documentation for the repository, which has details about how the codes may be run and also the current version of the live repository. Researchers should check this main README file to learn about current and subsequent version updates.

Gaze data from any source (e.g. eye tracking device, application or algorithm) must be formatted as per the instructions in the README files and used first with the main_proc.py in the "Data pre-processing" sub-folder to produce an output CSV file. Rest of the codes' functions are based on this output CSV file. Similarly, to use the GazeVisual GUI tool, users should copy and save the GazeVisual_v101.py file into any directory of their computer and run it as regular python codes, and make sure all the imported libraries are pre-installed.

Sample CSV files that can be used to test the codes are provided in each sub-folder of the repository for guidance. Input files format for the GUI tool is different and input data samples are included in the repository for users to understand the format. Links to sample videos showing the operation of the different windows of the GazeVisual GUI tool are in the "Sample videos" file of the GUI folder.

For the data repository on Mendeley, the data CSV files are labelled with the participant number, platform name and operating condition. They can be simply downloaded and the different column values in the files can be read to use or visualize them. A data documentation is also provided on the main page of the repository (as shown in Figure 5.5b, which appears when users visit the repository link provided in the beginning of this chapter).



(a) (b)

Figure 5.5: (a) View of GazeVisual-Lib code repository on GitHub (b) NUIG-Eyegaze01 data repository on Mendeley data.

5.4 Licensing information

GazeVisual-Lib is released as an open source repository under GNU-GPL v3.0 license which allows the users freedom to run, study, share and modify the software and the source codes in the repository. The users are requested to cite papers of Appendix A, D and G in their works.

The new gaze data repository is published under CC BY-NC 3.0 license. According to this license, Licensees may copy and distribute the material if they give the licensor the credits (attribution). Licensees may distribute derivative works only under a license identical to the license that governs the original work. The license also specifies that Licensees may use the data only for non-commercial purposes and there is also the condition that Licensees may copy, distribute, display and perform only verbatim copies of the work, not derivative works of it.

5.5 Utility and impact of open resources towards eye tracker and gaze data evaluation

The open code and data repositories developed and presented in this thesis are resources that would enable other researchers to explore and answer critical questions related to contemporary eye gaze research. For example, what are the performance limits and tolerances of a given eye tracker? How much (quantitatively) is an eye tracker's accuracy affected when operating under non ideal operating conditions? Which operating conditions affect the tracker's performance in a particular use-case? How can two gaze datasets, or the performance of two gaze estimation systems be compared quantitatively? What are the performance bottlenecks of individual gaze estimation algorithms? How can gaze error patterns be detected and predicted? The coding resources and the desktop application tool from the GazeVisual-Lib repository can help eye gaze researchers or engineer to find answers to these questions.

Since knowing the quality of gaze data is essential for ensuring the reliability of any gaze based application, the evaluation routines of the repository can be used to constantly monitor the data quality of any eye tracker, especially during practical operations under variable operating conditions.

Using the GazeVisual GUI, users can perform in depth gaze data evaluation without the need for detailed programming knowledge owing to its simple interface. This is particularly important due to the inter-disciplinary nature of gaze research where eye trackers are used widely by members from non-technological fields. The intended user group of GazeVisual-lib is quite diverse, ranging from developers of gaze estimation algorithms to researchers from fields like human-computer interaction, psychology and cognitive studies. Incidentally, gaze data quality is a critical aspect that affects all the stages of gaze data analysis, and the codes from GazeVisual-lib could be useful in this respect.

The experiments done as part of this thesis work have helped to develop and introduce a new eye gaze dataset which can aid in identifying the capabilities and limits of different eye tracking systems. Such labelled gaze datasets containing signatures of different operating conditions that frequently affect gaze data quality on different user platforms do not exist yet and keeping this in mind, the NUIG-Eyegaze01 dataset has been made available. The data can be put to a wide range of uses, including modelling and comparing error patterns, development and testing of gaze anomaly detection algorithms or gaze error compensation algorithms to name a few. The data can be put to a wide range of uses, including modelling and comparing gaze error patterns, development and testing of gaze anomaly detection algorithms or gaze error compensation methods to name a few. These are all sparsely explored areas in gaze research which could benefit from our open data repository.

The final and major significance of both the data and code repositories is that they are meant to encourage research towards practical and realistic performance evaluation of eye trackers, standardization of gaze research results and building of more open source tools for this purpose. The aim of presenting the open resources is to foster collaborative development and adoption of even better resources in this direction of gaze research, which ultimately can strengthen the usability and reliability of gaze estimation systems in their diverse range of applications.

Chapter 6

Discussions and future work

6.1 Summary and discussions

In this dissertation, several new strategies towards practical evaluation of generic commercial remote eye tracking systems and eye gaze data quality were proposed and developed. Firstly, through several literature reviews, the strong need for standardization and development of performance evaluation tools or protocols in contemporary gaze research were identified. This helped to lay the foundation for the later developmental parts of this thesis, which centred around the development of a performance evaluation framework for remote gaze estimation systems.
This framework provides a set of well-defined algorithms, software, datasets and experimental methods which can be used by the interdisciplinary eye gaze research and technology community for detailed evaluation of gaze data quality and eye tracking systems implemented on various applications, hardware or platforms.

The results from developing this evaluation framework include several new metrics and visualization methods for in-depth analysis of an eye tracking system's performance. The existing metrics that have been traditionally used for representing gaze estimation accuracy were found to be quite non-uniform and inadequate to express the impact of external conditions on gaze data or interpret underlying data patterns. Also previously, graphical representations of gaze accuracy or data quality variations were nearly non-existent.
The metrics developed as part of this work, as described in [13] and Chapter 4 (Section 4.2) are fully derived from raw gaze and ground truth data and may be used to quantitatively express the impact of several external conditions on gaze data characteristics from generic remote trackers. Some of these metrics may help in numerical comparison of data from multiple eye trackers or individuals while the ROC based metric proposed and developed in this thesis may be used to subjectively evaluate an eye tracker's performance, by setting user defined accuracy thresholds. The key benefit of these developments is that since full definitions of all these metrics are provided, these can be adopted

easily by any gaze researcher or system developer for their own gaze tracking systems. In addition, a set of visualizations are presented in Chapter 4, Section 4.3, which may be used to study gaze data quality, visually inspect for data anomalies or irregularities and summarize gaze data characteristics from multiple individuals and experiments. These are meant for quick and easy monitoring of an eye tracker's performance, especially during unconstrained operations.

Another result from this framework development comprise of software tools and a web-application which are presented in Chapter 4, Section 4.3. These were built to make the above mentioned metrics and visualizations easily accessible to generic users and gaze researchers. These tools are designed to provide standardized evaluation results using gaze data samples from an eye tracker as input, so that developers of any gaze based system or application may frequently monitor their gaze data quality without requiring specialized skills or programming knowledge.
It was also found that no such software tools or browser based applications for evaluating gaze data quality currently exist. This formed the motivation to put the software resources developed in this thesis work together in a GitHub code repository for use by the eye gaze research community, to increase the practical utility of the developed methods as well as to improve their outreach and scope for further development.

A new study was undertaken (described in Chapter 4, Section 4.4) as part of this thesis, by using the gaze datasets collected during the eye tracking experiments (described in Chapter 4, Section 4.1) for training machine learning models like SVM, KNN and neural networks. The purpose was to understand gaze error patterns produced by various "error sources" such as head pose, user distance and platform pose.
This study provided additional insights regarding gaze error characteristics and formed an important component of the evaluation framework, by including several methods for gaze data processing, augmentation, feature creation and classification. It was observed through this study that different error sources produce different gaze error distributions (in terms error magnitudes as well as spatial characteristics)and machine learning models are robust to detect the presence of different error sources in a complex gaze dataset where more than one source of error is present. Also regression models were built which could be used to predict gaze errors produced by two different error sources head pose and platform pose. Inputs for the prediction models were gaze angular variables and output was gaze error levels and it was observed that the ElasticNet model produced best prediction results.

Most of the materials developed as part of thesis have been released as open resources for the eye gaze community to use, extend and upgrade (discussed in Chapter 5). This includes the code repository described above containing the software implementation of gaze data evaluation metrics

and visualizations, and gaze data processing methods like outlier removal and augmentation with full documentation. Apart from this, the fully labelled gaze datasets, collected during this thesis work are made available for research use. The purpose of these open resources is to allow gaze researchers and system developers to use them to evaluate their gaze data in a homogeneous and standardized manner, and also use the gaze datasets for benchmark comparison of their gaze data quality. This is also intended to foster further and collaborative development towards standardization of gaze estimation systems and their outputs.

6.2 Future work

In this dissertation, the work was focussed on addressing the performance evaluation aspect of gaze estimation systems from multiple directions. However, as discussed before, gaze tracking is now a highly diverse and interdisciplinary field and the variety in eye tracking scenarios, operating platforms and system related issues is large. Therefore there still remain plenty of avenues for future work towards developing more sophisticated performance evaluation methods.

Firstly, the developed numerical metrics can be extended towards compound metrics which may be created by linear or non-linear combination of more than one metric. Examples are QoS or quality of service metrics which may include factors like noise, latency and gaze estimation accuracy to report performance of an eye tracker. QoS metrics are objective measures representing the overall performance of a system depending on several system-related characteristics. As discussed in Chapter 1, eye tracking has been implemented on several different use cases in consumer electronics, and thus QoS metrics may be developed for each use case based on their individual features. For example, for desktop systems, parameters determining QoS may be dependent on accuracy, head pose tolerance and visual angle dependence. For automotive systems, QoS may depend on accuracy, head pose and platform vibration tolerance, whereas for handheld devices, QoS could be a function of head pose and platform pose tolerance as well as accuracy.
Another direction could be developing entropy based metrics which would denote the degree of "order or disorder" in any given system and therefore can be used to observe the change in an eye tracker's system behaviour when any operating conditions are changed.

The dataset developed as part of this work could be extended to include more complex conditions such as when more than one error source are present. For example, gaze data may be collected when head pose and platform pose change simultaneously (e.g. in a tablet platform) or under the conditions of head pose variations and random platform vibrations(e.g. in an automotive platform).

Similarly, gaze data from head-mounted trackers under the simple and complex scenarios may be collected and compared with remote eye trackers, which may reveal unique, positive as well as problematic features in the gaze data from both these eye tracker types for different applications.

From the literature survey on publicly available gaze datasets, it was observed that there exists a deficit of gaze datasets for AR/VR systems, containing very near eye image frames and corresponding ground truths. Gaze data collected from an AR/VR setup under varied levels of head movement could be useful in studying gaze behaviours for these use cases. In addition, gaze data in virtual environments may be collected while a user performs a variety of tasks like walking or in a gaming scenario. These can then be used for identifying error sources arising in such unique use cases.

Apart from the experimental methods, there is a lot of scope to progress research on eye tracker performance evaluation through numerical modelling approaches. These could involve developing system models for different eye tracking use cases so that impact of component perturbations may be studied, building error compensation models for different operating conditions that may affect an eye tracker and using machine learning for unsupervised detection of gaze data anomalies (a part of which has been done in this thesis, discussed in Chapter 4, Section 4.4) to name a few.

Overall at this point on the roadmap of eye gaze research, realistic evaluation of gaze tracking systems and gaze data quality stands as a highly important milestone that needs to be covered through dedicated inter-disciplinary research in the days to come. This requires attention of researchers, developers, engineers and users alike, since gaze data quality, performance of eye trackers, usability of gaze information and advancement of gaze applications are all elements interconnected in a tight loop that will ultimately determine the future of eye gaze science and technology.

References

[1] C. H. Morimoto and M. R. M. Mimica, “Eye gaze tracking techniques for interactive applications,” Comput. Vis. Image Underst., vol. 98, no. 1, pp. 4–24, 2005.

[2] Robert J. K. Jacob. 1990. "What you look at is what you get: eye movement-based interaction techniques". i*n Proc.* SIGCHI Conference on Human Factors in Computing Systems (CHI '90), Jane Carrasco Chew and John Whiteside (Eds.). ACM, New York, NY, USA, 11-18.

[3] Florea, Laura, Corneliu Florea, Ruxandra Vrânceanu and Constantin Vertan. “Can Your Eyes Tell Me How You Think? A Gaze Directed Estimation of the Mental Activity.” i*n Proc. BMVC*(2013), pp 1-11.

[4] A. Hyrskykari, P. Majaranta, and K. J. Räihä, “From Gaze Control to Attentive Interfaces” *in Proc. HCII*, July, 2005.

[5] Duchowski, A.T. Behavior Research Methods, Instruments, & Computers (2002) 34: 455.

[6] A. Mohamed, M. Perreira da Silva, V. Courboulay. "A history of eye gaze tracking". Rapport Interne. 2007. HAL Id: hal-00215967 Available from: https://hal.archivesouvertes.fr/hal-00215967.

[7] P. M. Corcoran, F. Nanu, S. Petrescu, and P. Bigioi, “Real-time eye gaze tracking for gaming design and consumer electronics systems,” *IEEE Trans. Consum. Electron.*, vol. 58, no. 2, pp. 347–355, May 2012.

[8] V. Tanriverdi and R. K. Jacob. 2000."Interacting with eye movements in virtual environments". *in Proc*. SIGCHI conference on Human Factors in Computing Systems (CHI '00). ACM, New York, NY, USA, 265-272. vol. 2, no. 1, pp. 265–272, 2000.

[9] S. Bazrafkan, A. Kar, and C. Costache, “Eye Gaze for Consumer Electronics: Controlling and commanding intelligent systems.,” IEEE Consum. Electron. Mag., vol. 4, no. 4, pp. 65–71, 2015.

[10] K.Holmqvist, M.Nyström, and F. Mulvey. 2012. Eye tracker data quality: what it is and how to measure it. *in Proc*. of the Symposium on Eye Tracking Research and Applications (ETRA '12), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 45-52.

[11] K. Ooms, L. Lapon, L. Dupont, and S. Popelka, “Accuracy and precision of fixation locations recorded with the low-cost Eye Tribe tracker in different experimental set-ups,” *J. Eye Mov.*

Res., vol. 8, no. 1, pp. 1–24, 2015.

[12] A. Kar and P. Corcoran, "A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms," *in IEEE Access*, vol. 5, pp. 16495-16519, 2017.

[13] A. Kar; P. Corcoran, "Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations". *Sensors* **2018**, *18*, 3151.

[14] G. Funke, E. Greenlee, M. Carter, A. Dukes, R. Brown, and L. Menke, "Which eye tracker is right for your research? Performance evaluation of several cost variant eye trackers," in *Proc. Hum. Factors Ergon. Soc.*, vol. 60, no. 1, pp. 1240–1244, Sept. 2016.

[15] S.Popelka, Z. Stachoň, Č.Šašinka, and J. Doležalová. "EyeTribe Tracker Data Accuracy Evaluation and Its Interconnection with Hypothesis Software for Cartographic Purposes". Intell. Neuroscience 2016 (March 2016), 20.

[16] A. Gibaldi, M. Vanegas, P. J. Bex, and G. Maiello, "Evaluation of the Tobii EyeX Eye tracking controller and Matlab toolkit for research," *Behav. Res. Methods*, vol. 49, no. 3, pp. 923–946, 2017.

[17] X. L. C. Brolly and J. B. Mulligan, "Implicit Calibration of a Remote Gaze Tracker," 2004 Conference on Computer Vision and Pattern Recognition Workshop, Washington, DC, USA, 2004, pp. 134-134.

[18] T. Imabuchi, O. Dicky, A. Prima, H. Kikuchi, Y. Horie, and H. Ito, "Visible-spectrum Remote Eye Tracker for Gaze Communication," vol. 9443, no. ICGIP 2014, pp. 1–5, 2015.

[19] D. J. Mack, P. Schönle, S. Fateh, T. Burger, Q. Huang and U. Schwarz, "An EOG-based, head-mounted eye tracker with 1 kHz sampling rate," *in Proc.* IEEE Biomedical Circuits and Systems Conference (BioCAS), Atlanta, GA, 2015, pp. 1-4.

[20] D. Mazeika, A. Carbone, and E. E. Pissaloux, "Design of a generic head-mounted gaze tracker for human computer interaction," *Signal Process. - Algorithms, Archit. Arrange. Appl. Conf. Proceedings, SPA*, vol. 2015–April, pp. 127–131, 2015.

[21] J. Turner, A. Bulling, and H. Gellersen, "Extending the visual field of a head-mounted eye tracker for pervasive eye-based interaction," *in Proc. Symp. Eye Track. Res. Appl.*, vol. 1, no. 212, pp. 269–272, 2012.

[22] E. Schneider et al., "Gaze-aligned head-mounted camera with pan, tilt, and roll motion control for medical documentation and teaching applications," *in Proc.* IEEE International Conference on Systems, Man and Cybernetics, Taipei, 2006, pp. 327-331.

[23] K. Takemura, K. Takahashi, J. Takamatsu, and T. Ogasawara, "Estimating 3-D point-of-regard in a real environment using a head-mounted eye-tracking system," *IEEE Trans. Human-Machine Syst.*, vol. 44, no. 4, pp. 531–536, 2014.

[24] M. Y. Kim, S. Yang, and D. Kim, “Head-mounted binocular gaze detection for selective visual recognition systems,” *Sensors Actuators, A Phys.*, vol. 187, pp. 29–36, 2012.

[25] M. Meißner, et al., “Combining virtual reality and mobile eye tracking to provide a naturalistic experimental environment for shopper research”, *Journal of Business Research*, 2017.

[26] S. H. Lee, Jae-Young Lee and J. S. Choi, "Design and implementation of an interactive HMD for wearable AR system," *17th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, Ulsan, 2011, pp. 1-6..

[27] K. Choi, C. Ma and S. Ko, "Improving the usability of remote eye gaze tracking for human-device interaction," in *IEEE Transactions on Consumer Electronics*, vol. 60, no. 3, pp. 493-498, Aug. 2014.

[28] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, “Gaze tracking system at a distance for controlling IPTV,” *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[29] H. Drewes, R. Atterer, and A. Schmidt, “Detailed monitoring of user’s gaze and interaction to improve future e-learning,” *Univers. Access Hum. Comput. Interact. Ambient Interact. Proc. 4th Int. Conf. Univers. Access Hum. Comput. Interact. UAHCI2007, held as Part HCI Int. 2007*, pp. 802–811, 2007.

[30] A. Bulling, F. Alt, and A. Schmidt, “Increasing the security of gaze-based cued-recall graphical passwords using saliency masks,” *in Proc. 2012 ACM Annu. Conf. Hum. Factors Comput. Syst. - CHI ’12*, p. 3011, 2012.

[31] M. U. Ghani, S. Chaudhry, M. Sohail, and M. N. Geelani, “GazePointer: A real time mouse pointer control implementation based on eye gaze tracking,” *2013 16th Int. Multi Top. Conf. INMIC 2013*, pp. 154–159, 2013.

[32] B. B. Velichkovsky, M. a. Rumyantsev, and M. a. Morozov, “New Solution to the Midas Touch Problem - Identification of Visual Commands Via Extraction of Focal Fixations,” *Procedia Comput. Sci.*, vol. 39, pp. 75–82, 2014.

[33] Erroll Wood and Andreas Bulling. 2014. EyeTab: model-based gaze estimation on unmodified tablet computers. *in Proc.* Symposium on Eye Tracking Research and Applications (ETRA '14). ACM, New York, NY, USA, 207-210.

[34] V. Vaitukaitis and A. Bulling, “Eye gesture recognition on portable devices,” *in Proc. 2012 ACM Conf. Ubiquitous Comput. - UbiComp ’12*, p. 711, 2012.

[35] Xu, Junli et al. “Real-time eye tracking for the assessment of driver fatigue.” *Healthcare technology letters* vol. 5,2 54-58. 31 Jan. 2018, doi:10.1049/htl.2017.0020.

[36] I. H. Choi and Y. G. Kim, “Head pose and gaze direction tracking for detecting a drowsy driver,” *Appl. Math. Inf. Sci.*, vol. 9, no. 2, pp. 505–512, 2015.

[37] Q. Ji, Z. Zhu, and P. Lan, “Real-time nonintrusive monitoring and prediction of driver fatigue,” *IEEE Trans. Veh. Technol.*, vol. 53, no. 4, pp. 1052–1068, 2004.

[38] R. Zheng, K. Nakano, H. Ishiko, K. Hagita, M. Kihira and T. Yokozeki, "Eye-Gaze Tracking Analysis of Driver Behavior While Interacting With Navigation Systems in an Urban Area," in *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 4, pp. 546-556, Aug. 2016.

[39] D. W. Hansen and Q. Ji, "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 478-500, March 2010.

[40] C. Ma, K.-A. Choi, B.-D. Choi, and S.-J. Ko, “Robust remote gaze estimation method based on multiple geometric transforms,” *Opt. Eng.*, vol. 54, no. 8, p. 83103, 2015.

[41] K. Harezlak, P. Kasprowski, and M. Stasch, “Towards accurate eye tracker calibration -methods and procedures,” *Procedia Comput. Sci.*, vol. 35, no. C, pp. 1073–1081, 2014.

[42] Zhiwei Zhu and Qiang Ji, “Novel Eye Gaze Tracking Techniques Under Natural Head Movement,” *IEEE Trans. Biomed. Eng.*, vol. 54, no. 12, pp. 2246–2260, 2007.

[43] P. Blignaut, “Mapping the Pupil-Glint Vector to Gaze Coordinates in a Simple Video-Based Eye Tracker,” *J. Eye Mov. Res.*, vol. 7, no. 1, pp. 1–11, 2013.

[44] S. Shih, Y. Wu, C. Science, I. Engineering, and J. Liu, “A calibration-free gaze tracking technique,” *in proc. Int. Conf. Pattern Recognit.*, vol. 1, no. 6, pp. 201–204, 2000.

[45] K.Wang, S.Wang, and Q.Ji. 2016. "Deep eye fixation map learning for calibration-free eye gaze tracking." *in Proc.* Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16). ACM, New York, NY, USA, 47-55.

[46] T. Nagamatsu, J. Kamahara, and N. Tanaka. 2009. Calibration-free gaze tracking using a binocular 3D eye model. In CHI '09 Extended Abstracts on Human Factors in Computing Systems (CHI EA '09). ACM, New York, NY, USA, 3613-3618.

[47] K. Yamashiro, T. Takahashi, I. Ide, and H. Murase, “Automatic Calibration of an In-vehicle Gaze Tracking System Based on the Driver ’ s Gaze Behavior,” pp. 50–55, 2008.

[48] K. Yamashiro *et al*., "Automatic calibration of an in-vehicle gaze tracking system using driver's typical gaze behavior," *in Proc. IEEE Intelligent Vehicles Symposium*, Xi'an, 2009, pp. 998-1003.

[49] T. Santini, W. Fuhl, D. Geisler and E. Kasneci, "EyeRecToo: Open-source Software for Real-time Pervasive Head-mounted Eye Tracking". *in Proc.* 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2017), pages 96-101. DOI: 10.5220/0006224700960101.

[50] A. Poole and L. J. Ball, “Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects,” *Encycl. Human-Computer Interact.*, pp. 211–219, 2005.

[51] R. Zemblys, D. C. Niehorster, O. Komogortsev, and K. Holmqvist, “Using machine learning to

detect events in eye-tracking data," Behav Res 50: 160.,2018.

[52] G.Drusch, J. M. Christian Bastien, S.Paris. "Analysing eye-tracking data: From scanpaths and heatmaps to the dynamic visualisation of areas of interest", *in Proc.* International Conference on Applied Human Factors and Ergonomics, 2014, Krakow, Poland..

[53] I. Bacivarov, M. Ionita, and P. Corcoran, "Statistical models of appearance for eye tracking and eye-blink detection and measurement," *IEEE Trans. Consum. Electron.*, vol. 54, no. 3, pp. 1312–1328, 2008.

[54] Tobii Technology, "Accuracy and precision Test report Tobii T60 XL Eye tracker," *Technology*, 2011.

[55] T. E. Cognard, A. Goncharov, N. Devaney, C. Dainty and P. Corcoran, "A Review of Resolution Losses for AR/VR Foveated Imaging Applications," *in Proc. 2018 IEEE Games, Entertainment, Media Conference (GEM)*, Galway, 2018, pp. 1-9.

[56] A. Kar and P. Corcoran, "Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms," *in Proc 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, 2016, pp. 002061-002066.

[57] F. L. Coutinho and C. H. Morimoto, "Free head motion eye gaze tracking using a single camera and multiple light sources," *in Proc.,19th Brazilian Symp. Comput. Graph. Image Process.*, pp. 171–178, 2006.

[58] H. Drewes, A. De Luca, and A. Schmidt, "Eye-gaze interaction for mobile phones," *Mobil. '07 in proc. 4th Int. Conf. Mob. Technol. Appl. Syst. 1st Int. Symp. Comput. Hum. Interact. Mob. Technol.*, vol. 7, pp. 364–371, 2007.

[59] J.M. Bosten, G. Bargary, P.T**.** Goodbourn, R.E. Hogg, A.J. Lawrance-Owen, & J.D Mollon. "Individual differences provide psychophysical evidence for separate on- and off-pathways deriving from the short-wave cones." *Journal of the Optical Society of America A*, *31*(4), A47–54.

[60] J. Lemley, A. Kar and P. Corcoran, "Eye Tracking in Augmented Spaces: A Deep Learning Approach," *in Proc, 2018 IEEE Games, Entertainment, Media Conference (GEM)*, Galway, 2018, pp. 1-6.

[61] E. M. Reingold, "Eye tracking research and technology: Towards objective measurement of data quality," *Vis. cogn.*, vol. 22, no. 3, pp. 635–652, 2014.

[62] A. Kar, S. Bazrafkan, C. Costache and P. Corcoran, "Eye-gaze systems - An analysis of error sources and potential accuracy in consumer electronics use cases," *in proc. 2016 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, 2016, pp. 319-320..

[63] A. Kar and P. Corcoran, "Gaze Visual - A Graphical Software Tool for Performance Evaluation

of Eye Gaze Estimation Systems," *in proc. 2018 IEEE Games, Entertainment, Media Conference (GEM)*, Galway, 2018, pp. 1-9.

[64] J. Lemley, A. Kar, A. Drimbarean and P. Corcoran, "Convolutional Neural Network Implementation for Eye-Gaze Estimation on Low-Quality Consumer Imaging Systems," in *IEEE Transactions on Consumer Electronics*. doi: 10.1109/TCE.2019.2899869

[65] R. S. Remmel, "An Inexpensive Eye Movement Monitor Using the Scleral Search Coil Technique," in *IEEE Transactions on Biomedical Engineering*, vol. BME-31, no. 4, pp. 388-390, April 1984.

[66] Eric Whitmire, Laura Trutoiu, Robert Cavin, David Perek, Brian Scally, James Phillips, and Shwetak Patel. 2016. EyeContact: scleral coil eye tracking for virtual reality. In Proc. 2016 ACM International Symposium on Wearable Computers (ISWC '16). ACM, New York, NY, USA, 184-191.

[67] J. A Müller, D. Wendt, B. Kollmeier, T. Brand"Comparing Eye Tracking with Electrooculography for Measuring Individual Sentence Comprehension Duration". PLoS ONE 11(10): e0164627.

[68] H.C. Lee, W.O Lee; C.W. Cho.; S.Y. Gwon; K.R. Park; H.. Lee, J .Cha. "Remote Gaze Tracking System on a Large Display". *Sensors* **2013**, *13*, 13439-13463.

[69] S.-W. Shih and J. Liu, "A novel approach to 3-D gaze tracking using stereo cameras.," *in Proc. IEEE Trans. Syst. Man. Cybern. B. Cybern.*, vol. 34, no. 1, pp. 234–245, 2004.

[70] M. Kassner, W.Patera, and A.Bulling. "Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction". In Proc. 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication (UbiComp '14 Adjunct). ACM, New York, NY, USA, 1151-1160.

[71] D. Vitonis and D. W. Hansen, "Person Identification using Eye Movements and Post Saccadic Oscillations," *in Proc Tenth Int. Conf. Signal-Image Technol. Internet-Based Syst.*, pp. 580–583, 2014.

[72] Y. Zhang and M. Juhola, "On Biometrics With Eye Movements," in *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 5, pp. 1360-1366, Sept. 2017.

[73] P. Kasprowski, J. Ober, "Eye Movements in Biometrics". In: Maltoni D., Jain A.K. (eds) Biometric Authentication. BioAW 2004. Lecture Notes in Computer Science, vol 3087. Springer, Berlin, Heidelberg.

[74] I. Rigas and G. Economou, "Human eye movements as a trait for biometrical identification," *in Proc. 2012 IEEE Fifth Int. Conf. Biometrics Theory, Appl. Syst.*, pp. 217–222, 2012.

[75] B. A. Smith, Q. Yin, S. K. Feiner, and S. K. Nayar. "Gaze locking: passive eye contact detection for human-object interaction". *In* Proc. 26th annual ACM symposium on User interface software

and technology (UIST '13). ACM, New York, NY, USA, 271-280.

[76] U. Weidenbacher, G. Layher, P. Strauss and H. Neumann, "A comprehensive head pose and gaze database," *2007 3rd IET International Conference on Intelligent Environments*, Ulm, 2007, pp. 455-458.

[77] C.D. McMurrough, V. Metsis, J. Rich, and F. Makedon. "An eye tracking dataset for point of gaze detection." *In Proc.* Symposium on Eye Tracking Research and Applications (ETRA '12), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 305-308.

[78] Y. Sugano, Y. Matsushita, and Y. Sato, "Learning-by-synthesis for appearance-based 3D gaze estimation," *in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1821–1828, 2014.

[79] X. Zhang, Y.Sugano, M.Fritz, and A.Bulling. "MPIIGaze: RealWorld Dataset and Deep Appearance-Based Gaze Estimation" IEEE Transactions on Pattern Analysis and Machine Intelligence (2018).

[80] H. Q. et al."OMEG: Oulu Multi-Pose Eye Gaze Dataset". In: Paulsen R., Pedersen K. (eds) Image Analysis. SCIA 2015. Lecture Notes in Computer Science, vol 9127. Springer, Cham.

[81] N. Li and C.Busso. "Evaluating the robustness of an appearance-based gaze estimation method for multimodal interfaces", *in Proc.* 15th ACM on International conference on multimodal interaction (ICMI '13). ACM, New York, NY, USA, 91-98. 2013.

[82] K.Funes Mora, F.Monay, and J. Odobez. 2014. "EYEDIAP: a database for the development and evaluation of gaze estimation algorithms from RGB and RGB-D cameras." i*n Proc* (ETRA '14). ACM, New York, NY, USA, 255-258.

[83] N. Erdogmus and S. Marcel, "Spoofing in 2D face recognition with 3D masks and anti-spoofing with Kinect," i*n Proc 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, Arlington, VA, 2013, pp. 1-6.

[84] S.Asteriadis, D.Soufleros, K. Karpouzis, and S.Kollias. "A natural head pose and eye gaze dataset." *in* Proc. AFFINE '09. ACM, New York, NY, USA, Article 1 , 4 pages.

[85] A.Villanueva, V. Ponz, L. Sesma-Sanchez, M. Ariz, S. Porta, and R. Cabeza. "Hybrid method based on topography for robust detection of iris center and eye corners." ACM Trans. Multimedia Comput. Commun. Appl. 9, 4, Article 25 (August 2013), 20 pages.

[86] C. C. Lai, Y. T. Chen, K. W. Chen, S. C. Chen, S. W. Shih, and Y. P. Hung, "Appearance-based gaze tracking with free head movement," *in Proc. - Int. Conf. Pattern Recognit.*, vol. 1, pp. 1869–1873, 2014.

[87] I. F. Ince and J. W. Kim, "A 2D eye gaze estimation system with low- resolution webcam images," *EURASIP J. Adv. Signal Process.*, vol. 2011, no. 1, p. 40, 2011.

[88] Y.-L. Wu, C.-T. Yeh, W.-C. Hung, and C.-Y. Tang, "Gaze direction estimation using support

vector machine with active appearance model," *Multimed. Tools Appl.*, pp. 1–26, 2012.

[89] R. Oyini Mbouna, S. G. Kong and M. Chun, "Visual Analysis of Eye State and Head Pose for Driver Alertness Monitoring," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1462-1469, Sept. 2013.

[90] J. H. Oh and N. Kwak, "Recognition of a Driver's gaze for vehicle headlamp control," *IEEE Trans. Veh. Technol.*, vol. 61, no. 5, pp. 2008–2017, 2012.

[91] S. J. Lee, J. Jo, H. G. Jung, K. R. Park, and J. Kim, "Real-time gaze estimator based on driver's head orientation for forward collision warning system," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 254–267, 2011.

[92] X. Sun, L. Xu, and J. Yang, "Driver fatigue alarm based on eye detection and gaze estimation," *in Proc. SPIE*, vol. 6786, no. 1, pp. 678612-678612–6, 2007.

[93] Y. Liang, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 340–350, 2007.

[94] T.W. Victor, J.L. Harbluk, J. Engström, "Sensitivity of eye-movement measures to in-vehicle task difficulty", Transportation Research Part F: Traffic Psychology and Behaviour, 8 (2) (2005), pp. 167-190

[95] H. Hadizadeh, M. J. Enriquez and I. V. Bajic, "Eye-Tracking Database for a Set of Standard Video Sequences," in *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 898-903, Feb. 2012.

[96] Rajashekar, U. van der Linde, I. Bovik, A.C. Cormack, L.K. "Statistical analysis and selection of visual fixations", *Journal of Vision*, 6(6):496, 496a, .

[97] C. Koch, et al, "*Predicting human gaze using low-level saliency combined with face detection".* Advances in Neural Information Processing Systems , 20 . pp. 1-7.

[98] G. Kootstra, et al. "Predicting Eye Fixations on Complex Visual Stimuli Using Local Symmetry." *Cognitive computation* vol. 3,1 (2011): 223-240.

[99] J. Li, M. D. Levine, X. An, X. Xu and H. He, "Visual Saliency Based on Scale-Space Analysis in the Frequency Domain," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 996-1010, April 2013.

[100] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba and F. Durand, "What Do Different Evaluation Metrics Tell Us About Saliency Models?," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 3, pp. 740-757, 1 March 2019.

[101] Judd, T. Durand, F. Torralba, A. (2010). Fixations on Low Resolution Images. *Journal of Vision*, 10(7):142, 142a.

[102] S. Ramanathan, H. Katti, N. Sebe, M. Kankanhalli, TS. Chua. "An Eye Fixation Database for Saliency Detection in Images." In: Daniilidis K., Maragos P., Paragios N. (eds) Computer

Vision – ECCV 2010. ECCV 2010. Lecture Notes in Computer Science, vol 6314. Springer, Berlin, Heidelberg

[103] N. Bruce, J. Tsotsos, "Attention based on information maximization". *Journal of Vision*, 7(9):950, 950a, (2007).

[104] U. Engelke and A. Maeder, "Visual Attention Modelling for Subjective Image Quality Databases," *in Proc. 2009 IEEE Int. Work. Multimed. Signal Process.*, pp. 1–6.

[105] S. Mathe and C. Sminchisescu, "Actions in the Eye: Dynamic Gaze Datasets and Learnt Saliency Models for Visual Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1408-1424, 1 July 2015.

[106] M. Jiang, S. Huang, J. Duan and Q. Zhao, "SALICON: Saliency in Context," *in Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015, pp. 1072-1080.

[107] A. Palazzi, D. Abati, S. Calderara, F. Solera and R. Cucchiara, "Predicting the Driver's Focus of Attention: the DR(eye)VE Project," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[108] J.Xu, M.Jiang, S.Wang, M.Kankanhalli, and Qi Zhao, "Predicting Human Gaze Beyond Pixels," in Journal of Vision, Volume 14, Issue 1, Article 28, Pages 1-20, January 2014. .

[109] N. F. Duarte, M. Raković, J. Tasevski, M. I. Coco, A. Billard and J. Santos-Victor, "Action Anticipation: Reading the Intentions of Humans and Robots," in *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4132-4139, Oct. 2018.

[110] K. S. Lohan, E. Sheppard, G. Little, and G. Rajendran, "Toward Improved Child – Robot Interaction by Understanding Eye Movements," *IEEE Trans. Cogn. Dev. Syst.*, vol. 10, no. 4, pp. 983–992, 2018.

[111] R.Cambuzat, F. Elisei, G.Bailly, O. Simonin, A.Spalanzani. "Immersive Teleoperation of the Eye Gaze of Social Robots Assessing Gaze-Contingent Control of Vergence, Yaw and Pitch of Robotic Eyes." *in Proc. ISR 2018 - 50th International Symposium on Robotics*, Jun 2018, Munich, Germany. pp.232-239, 2018, .

[112] K.Sengupta, M.Ke, R.Menges, C.Kumar, and S.Staab. "Hands-free web browsing: enriching the user experience with gaze and voice modality." In Proc. 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18). ACM, New York, NY, USA, Article 88, 3 pages..

[113] Y.Gizatdinova, O.Špakov, O.Tuisku, M.Turk, and V.Surakka. "Gaze and head pointing for hands-free text entry: applicability to ultra-small virtual keyboards." i*n Proc.* 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18). ACM, New York, NY, USA, Article 14, 9 pages..

[114] B. Masse, S. Ba, and R. Horaud, "Tracking Gaze and Visual Focus of Attention of People

Involved in Social Interaction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 11, pp. 2711–2724, 2018.

[115] D.C. Niehorster, T. Cornelissen, K. Holmqvist, K. et al. Atten Percept Psychophys (2019) 81: 666.

[116] S. Grogorick, G. Albuquerque and M. Maqnor, "Gaze Guidance in Immersive Environments," *in Proc. 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, Reutlingen, 2018, pp. 563-564.

[117] P. Lungaro, F. Saeik, and K. Tollmar. "Demonstration of gaze-aware video streaming solutions for mobile VR." *in Proc.* ACM SIGGRAPH 2018 Virtual, Augmented, and Mixed Reality (SIGGRAPH '18). ACM, New York, NY, USA, Article 14, 1 pages.

[118] T. E. Cognard, C. Dainty, and N. Devaney, "Estimating axial resolution with diffraction theory," Appl. Opt. 57, E138-E141 (2018).

[119] J. Pettersson, A. Albo, J. Eriksson, P. Larsson, K. W. Falkman, and P. Falkman, "Cognitive Ability Evaluation using Virtual Reality and Eye Tracking," *2018 IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl.*, pp. 1–6, 2018.

[120] X. Fan, F. Wang, Y. Lu, D. Song and J. Liu, "Eye Gazing Enabled Driving Behavior Monitoring and Prediction," *in Proc. 2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, San Diego, CA, 2018, pp. 1-4.

[121] S. Jha and C. Busso, "Probabilistic Estimation of the Gaze Region of the Driver using Dense Classification", *in Proc.2018 21st Int. Conf. Intell. Transp. Syst.*, pp. 697–702, 2018.

[122] Henrik Skovsgaard, John Paulin Hansen, and Emilie Møllenbach. 2013. Gaze tracking through smartphones. In Gaze Interaction in the Post-WIMP World CHI 2013 One-day Workshop..

[123] A. Vora, R. Shah, H. Pottekkatt and D. Parmar, "Gaze-contingent Screen Adjustment," *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, Coimbatore, 2018, pp. 565-568..

[124] S. Wibirama and H. A. Nugroho, "Towards Understanding Addiction Factors of Mobile Devices : An Eye Tracking Study on Effect of Screen Size," *2017 39th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, pp. 2454–2457, 2017.

[125] Brousseau, B.; Rose, J.; Eizenman, M. Accurate Model-Based Point of Gaze Estimation on Mobile Devices.

[126]Ion Martinikorena, Rafael Cabeza, Arantxa Villanueva, Sonia Porta: Introducing I2head database. PETMEI@ETRA 2018: 1:1-1:7

[127] K. Ruhland, et al 2015. A Review of Eye Gaze in Virtual Agents, Social Robotics and HCI: Behaviour Generation, User Interaction and Perception. Comput. Graph. Forum 34, 6 (September 2015), 299-326..

[128] J. F. Blinn, "A trip down the graphics pipeline: grandpa, what does 'viewport' mean?," in IEEE Computer Graphics and Applications, vol. 12, no. 1, pp. 83-87, Jan. 1992 .

[129] Z. Zhou and P. Tang, "Continuous anomaly detection in satellite image time series based on Z-scores of Season-Trend model Residuals," *in Proc. 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, 2016, pp. 3410-3413.

[130] W. Jia, H. Zhang, X. He and Q. Wu, "A Comparison on Histogram Based Image Matching Methods," *2006 IEEE International Conference on Video and Signal Based Surveillance*, Sydney, Australia, 2006, pp. 97-97..

[131] J. Qiang-rong and G. Yuan, "Face recognition based on Detail Histogram Intersection kernel," *in Proc. IEEE International Conference on Intelligent Computing and Intelligent Systems*, Shanghai, 2009, pp. 71-74.

[132] A. Khatoon and P. Corcoran, "Android permission system and user privacy — A review of concept and approaches," *in Proc.2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, Berlin, 2017, pp. 153-158.

[133] T. Chen, K.Ma and L.Chen, "Tri-state median filter for image denoising," in *IEEE Transactions on Image Processing*, vol. 8, no. 12, pp. 1834-1838, Dec. 1999.

[134] T. Khalil, "A Survey of Feature Selection and Feature Extraction Techniques in Machine Learning," *2014 Sci. Inf. Conf.*, pp. 372–378, 2014.

[135] T. Verma, A. P. S. Tiwana, and C. C. Reddy, "Data Analysis to Generate Models Based on Neural Network and Regression for Solar Power Generation Forecasting," *inPproc 2016 7th Int. Conf. Intell. Syst. Model. Simul.*, pp. 97–100, 2016.

[136] Okfalisa, I. Gazalba, Mustakim and N. G. I. Reza, "Comparative analysis of k-nearest neighbor and modified k-nearest neighbor algorithm for data classification," *2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Yogyakarta, 2017, pp. 294-298.

[137] Polson, Nicholas G.; Scott, Steven L. Data augmentation for support vector machines. Bayesian Anal. 6 (2011), no. 1, 1--23. doi:10.1214/11-BA601..

[138] R. Bayindir, M. Ieee, M. Gok, and A. Dataset, "An Intelligent Power Factor Correction Approach Based on Linear Regression and Ridge Regression Methods," *in Proc. 2011 10th Int. Conf. Mach. Learn. Appl. Work.*, vol. 2, pp. 313–315, 2011.

[139] S. D. Iyer and H. Ramasangu, "Hybrid LASSO and Neural Network estimator for gaze estimation," *in Proc.IEEE Region 10 Conference (TENCON)*, Singapore, 2016, pp. 2579-2582..

[140] T.T. Um, et al. "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks." *in Proc.* 19th ACM International Conference on Multimodal Interaction (ICMI '17). ACM, New York, NY, USA, 216-220.

Appendix A: A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 21, 2017, accepted July 23, 2017, date of publication August 7, 2017, date of current version September 6, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2735633

A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms

ANURADHA KAR, (Student Member, IEEE), AND PETER CORCORAN, (Fellow, IEEE)
Center for Cognitive, Connected, and Computational Imaging, Department of Electrical and Electronic Engineering, National University of Ireland, Galway, Ireland

Corresponding author: Anuradha Kar (a.kar2@nuigalway.ie)

This work was supported in part by the Science Foundation Ireland through the Strategic Partnership Program and in part by FotoNation Ltd., Next Generation Imaging for Smartphone and Embedded Platforms, under Project 13/SPP/I2868.

ABSTRACT In this paper, a review is presented for the research on eye gaze estimation techniques and applications, which has progressed in diverse ways over the past two decades. Several generic eye gaze use-cases are identified: desktop, TV, head-mounted, automotive, and handheld devices. Analysis of the literature leads to the identification of several platform specific factors that influence gaze tracking accuracy. A key outcome from this review is the realization of a need to develop standardized methodologies for the performance evaluation of gaze tracking systems and achieve consistency in their specification and comparative evaluation. To address this need, the concept of a methodological framework for practical evaluation of different gaze tracking systems is proposed.

INDEX TERMS Eye gaze, gaze estimation, accuracy, error sources, performance evaluation, user platforms.

I. INTRODUCTION

Advances in eye gaze tracking technology over the past few decades have led to the development of promising gaze estimation techniques and applications for human computer interaction. Historically, research on gaze tracking dates back to the early 1900s, starting with invasive eye tracking techniques. These included electro-occulography using pairs of electrodes placed around the eyes or the scleral search methods that include coils embedded into a contact lens adhering to the eyes. The first video based eye tracking study was made on pilots operating airplane controls in the 1940s [1]. Research on head-mounted eye trackers advanced in the 1960s and gaze tracking developed further in the 1970s with focus on improving accuracy and reducing the constraints on users. With increasing computing power in devices, real time operation of eye trackers became possible during the 1980s. However till this time, owing to limited availability of computers, eye tracking was mainly limited to psychological and cognitive studies and medical research. The application focus towards general purpose human computer interaction was sparse. This changed in the 1990s as eye gaze found applications in computer input and control [2]. Post 2000, rapid advancements in computing speed, digital video processing and low cost hardware brought gaze tracking equipment closer to users, with applications in gaming, virtual reality and web-advertisements [3].

Eye gaze information is used in a variety of user platforms. The main use cases may be broadly classified into (i) desktop computers [4]–[6], (ii) TV panels [7], [8], (iii) head mounted [9]–[12] (iv) automotive setups [13]–[17] (v) handheld devices [18], [19]. Applications based on desktop platforms involve using eye gaze for computer communication and text entry, computer control and entering gaze based passwords [20]. Remote eye tracking has recently been used on TV panels to achieve gaze controlled functions, for example selecting and navigating menus and switching channels. Head-mounted gaze tracking setups usually comprise of two or more cameras mounted on a support framework worn by the user. Such systems have been extensively employed in user attention and cognitive studies, psychoanalysis, oculomotor measurements [2], virtual and augmented reality applications [21], [22]. Real time gaze and eye state tracking on automotive platforms is used in driver support systems to evaluate driver vigilance and drowsiness levels. These use eye tracking setups mounted on a car's dashboard along with computing hardware running machine vision algorithms. In handheld devices such as smartphones or tablets, the front camera is used to track user gaze to activate functions such

as locking/unlocking phones, interactive displays, dimming backlights or suspending sensors [18], [23].

Within each of these use cases there exists a wide range of system configurations, operating conditions and varying quality of imaging and optical components. Furthermore, the variations in eye-movement and biological aspects of individuals lead to challenges in achieving consistent and repeatable performance from gaze tracking methods. Thus, despite several decades of development in eye gaze research, performance evaluation and comparison of different gaze estimation techniques across different platforms is a still a difficult task [24], [25].

In order to provide insight into the current status of eye gaze research and outcomes, this paper presents a detailed literature review and analysis that considers algorithms, system configuration, user conditions and performance issues for existing gaze tracking systems. Specifically, use-cases based on five different eye gaze platforms are considered.

The aim of this work is to gain a realistic overview of the diversity currently existing in this field and to identify the factors that affect the practical usability of gaze tracking systems. Further, this review highlights the need for developing standardized measurement protocols to enable evaluation and comparison of the performance and operational characteristics of different gaze tracking systems. In this paper, first the diversity and standardization issues in different aspects of eye gaze research are discussed and then the idea of a performance evaluation framework is proposed. This framework includes several planned and ongoing experiments that are aimed at practical evaluation of any gaze tracker. Our goal is to encourage further discussion and additional contributions from researchers in this field.

There have been detailed review works on eye gaze made in the last few years such as [3] and [26]–[28] which discussed about recent developments in gaze tracking methods, comparing different estimation techniques, setups, applications and challenges involved in using gaze as an input modality. Hansen and Ji [26] provides an in-depth review on different eye models, eye detection techniques and models for gaze estimation, along with a summary of gaze applications. It also discusses inaccuracies in gaze tracking arising from the eye model components, jitter and refraction due to user wearing glasses. However, our work differs on several grounds from these reviews. Firstly, our review is specifically aimed towards highlighting the issues affecting realistic performance evaluation of gaze tracking systems, such as ambiguous accuracy metrics and un-accounted error sources. Secondly, we do a detailed classification of four different eye gaze research platforms. Extensive literature resources are collected and analyzed for each platform with the aim to understand the factors that affect the performance of a gaze based system in each of these. Thirdly, we present our survey in a statistical format that shows the lack of standardization in eye gaze research as a quantitative observation. Also our survey includes research works published until 2017 to make it exhaustive and up-to-date. Finally, our review not only provides an overview of the current status of eye gaze research but also forms the foundation of a performance evaluation framework for eye gaze systems which is proposed by us in Section VI of this work and is currently under development.

The paper is organized as follows: Section II presents a brief overview on eye movements, gaze tracking systems and accuracy measures used in contemporary gaze research. In Sections III and IV, several gaze tracking algorithms are categorized and key research works on the implementation of gaze tracking in five different user platforms are reviewed. In Section V, the factors limiting practical performance of gaze tracking in different user platforms are analyzed and issues with diversity in gaze accuracy metrics are discussed. The background and concept of a methodological framework for practical evaluation of eye gaze systems is presented in Section VI. We note here that the scope of this review excludes gaze tracking for clinical and neurological directions, retinal imaging and studies on children and patients.

II. EYE GAZE TRACKING FUNDAMENTALS

A. TYPES OF EYE MOVEMENTS STUDIED

Several types of eye movements are studied in eye gaze research and applications to collect information about user intent, cognitive processes, behavior and attention analysis [28]–[31]. These are broadly classified as follows: 1. Fixations: These are phases when the eyes are stationary between movements and visual input occurs. Fixation related measurement variables include total fixation duration, mean fixation duration, fixation spatial density, number of areas fixated, fixation sequences and fixation rate. 2. Saccades: These are rapid and involuntary eye movements that occur between fixations. Measurable saccade related parameters include saccade number, amplitude and fixation-saccade ratio 3. Scanpath: This includes a series of short fixations and saccades alternating before the eyes reach a target location on the screen. Movement measures derived from scanpath include scanpath direction, duration, length and area covered 4. Gaze duration: It refers to the sum of all fixations made in an area of interest before the eyes leave that area and also the proportion of time spent in each area. 5. Pupil size and blink: Pupil size and blink rate are measures used to study cognitive workload. Table 1 presents the characteristics of different eye movements and their applications.

B. BASIC SETUP AND METHOD USED FOR EYE GAZE ESTIMATION

Video based eye gaze tracking systems comprise fundamentally of one or more digital cameras, near infra-red (NIR) LEDs and a computer with screen displaying a user interface where the user gaze is tracked. A typical eye gaze tracking setup is shown in Fig. 1. The steps commonly involved in passive video based eye tracking include user calibration, capturing video frames of the face and eye regions

TABLE 1. Classification of eye movements.

Eye movement type	Movement rate	Latency/duration of occurrence	Functionality/ Significance	Applications in Human Computer interaction
Fixation	< 15 – 100 deg/ms	180- 275 ms	Acquiring information, Cognitive processing, attention	Browsing information, reading, scene perception
Saccade	100- 700 deg/sec	Latency-200 ms, duration: 20–200 ms	Moving between targets	Visual search
Smooth pursuit	< 100 deg/sec based on target speed	100 ms	Following moving targets	Gaze based drawing, steering
Scanpath	--	--	Scanning, direct search	Assessing user behavior, user interface and layout quality
Gaze duration	--	--	Cognitive processing, conveying intent	Item selection, text/number entry
Blink	12 -15 per min	300 ms	Indicates behavioral states, stress	Eye liveliness detection, activate command/control
Pupil size change	4-7 mm/sec	140 ms	Cognitive effort, representing micro-emotions	Assessing cognitive workload, user fatigue, command/control

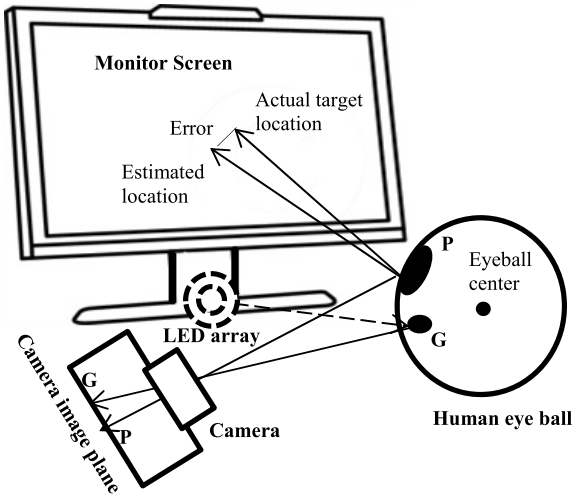


FIGURE 1. Schematic diagram of a typical gaze tracking system. P is the pupil of the human eye ball and G is the glint location formed on the cornea, which are imaged on the camera plane. The figure also shows the error in gaze estimation as the deviation between actual & estimated gaze locations.

of user, eye detection and mapping with gaze coordinates on screen. The common methodology (called Pupil Center Corneal Reflection or PCCR method) involves using NIR LEDs to produce glints on the eye cornea surface and then capturing images/videos of the eye region [26], [32]. Gaze is estimated from the relative movement between the pupil center and glint positions. External NIR illumination with single/multiple LEDs (wavelengths typically in the range 850+/− 30 nm with some works such as [33] using 940 nm) is often used to achieve better contrast and avoid effects due to variations induced by natural light. Webcams are mostly used; those operate at 30/60 fps frame rate and have infrared (IR) transmission filters to block out the visible light. Different gaze tracking methods are discussed in detail in Section III.

The user-interface for gaze tracking can be active or passive, single or multimodal [34]–[36]. In an active user interface, the user's gaze can be tracked to activate a function and gaze information can be used as an input modality. A passive interface is a non-command interface where eye gaze data is collected to understand user interest or attention. Single modal gaze tracking interfaces use gaze as the only input variable whereas a multimodal interface combines gaze input along with mouse, keyboard, touch, or blink inputs for command.

C. CALIBRATION

A generalized structure and model of a human eye is shown in Fig. 4a. The eye parameters typically required in gaze estimation are pupil center, center of curvature of cornea, the optical and the visual axes [32]. The posterior of the eyeball is called retina and the center of the retina with highest visual sensitivity is called the Fovea. The line joining the fovea with the center of corneal curvature is called the visual axis. Optical axis is the line passing through the pupil center and center of corneal curvature as shown in Fig 4a. The visual axis determines the direction of gaze and deviates from the optical axis. This offset is known as the kappa angle and measures around 5 degrees [37] but is dependent on each user. In gaze estimation, the pan and tilt components of the kappa angle are unique to each user and the visual axis cannot be estimated directly. The visual axis and the kappa angle therefore have to be obtained for each person though a process called calibration which has to be done at the start of an eye tracking procedure. Calibration is performed by showing the user a set of specific targets distributed over the front screen (as shown in Fig. 2) and the user is asked to gaze at them for a certain amount of time [38]. The tracker camera captures the various eye positions for each target point which are then mapped to the corresponding gaze coordinates and thus the tracker learns this mapping function. Calibration routines differ in the number and layout of target points, user fixation duration at each point and type of mapping algorithm used.

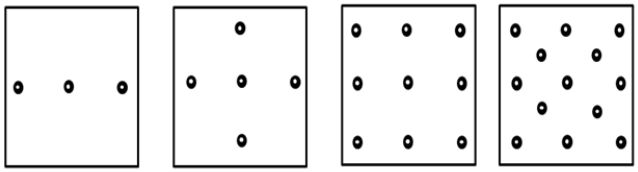


FIGURE 2. Calibration screen with 3, 5, 9, 13 target points.

D. CORRESPONDENCE OF EYE GAZE WITH HEAD POSITIONS

The gaze location of a user depends both on the gaze direction and also on the head orientation [39]. In methods which use PCCR techniques, if the user moves their head with respect to the tracker-camera axis while looking at the same point on the front screen, the glint vectors with respect to the pupil centers (for two different eye locations produced by head movement) will be different from each other. Therefore the estimated gaze locations will be inaccurate. Eqn 1 presents the relationship between user reference gaze directions (d_{kref}), head pose direction d_k and actual gaze direction (d_{kgaze}) which is a result of both head and eye rotation, as shown in Fig. 3. The effect of head movement has to be compensated before applying the gaze mapping algorithm or a chin rest for fixing head pose has to be used.

$$d_k - d_{kref} = k\left(d_{kgaze} - d_{kref}\right) \tag{1}$$

where k is a parameter related to head pan and tilt reported in [39] with values 0.5 and 0.4 respectively.

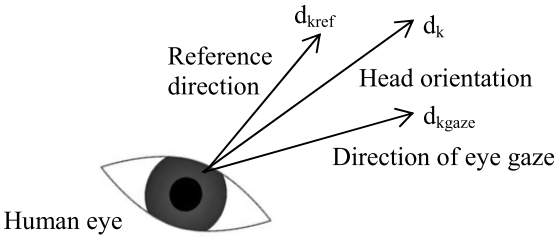


FIGURE 3. Relation between gaze direction & head pose.

E. ESTIMATION OF GAZE TRACKING ACCURACY

In a typical eye gaze tracking operation, a user gazes at an interface on a computer screen which provides them with visual stimulus in the form of a set of targets or a scene. Gaze tracking accuracy is estimated as the average difference between the real stimuli positions and the measured gaze positions, which also provides an idea about the performance of the system.

In conventional literature gaze tracking accuracy measures are presented in different ways e.g. angular accuracy in degrees, distance accuracy in cm or distances in pixels. These accuracy estimate calculations are shown below. In practice, calculations are made separately for both eyes. For brevity, single calculations are presented and the same equation holds for both right and left eye. $POG{\cdot}X_{left}$, $POG{\cdot}Y_{left}$, $POG{\cdot}X_{right}$, $POG \cdot Y_{right}$ are the measured X,Y coordinates of the left and right eye's point of gaze (PoG). The mean gaze coordinates considering both eyes are $POG \cdot X$ *and* $POG \cdot Y$. *dist* is the distance of the eye from the screen and *mean_dist* is the mean distance of eye from the tracker. The x/y pixels are the pixel shifts in x/y directions and offset is the distance between the tracker sensor and lower edge of display screen. Further details on these calculations can be found in [40].

Gaze point coordinates:

$$\mathrm{POG \cdot X} = \mathrm{mean}\left(\frac{\mathrm{POG \cdot X_{left}} + \mathrm{POG \cdot X_{right}}}{2}\right) \tag{2}$$

$$\mathrm{POG \cdot Y} = \mathrm{mean}\left(\frac{\mathrm{POG \cdot Y_{left}} + \mathrm{POG \cdot Y_{right}}}{2}\right) \tag{3}$$

Pixel accuracy (Pix_acc):

$$\mathrm{Pix\_acc} = \sqrt{\left((\mathrm{target \cdot X} - \mathrm{POG \cdot X})^2 + (\mathrm{target \cdot Y} - POG \cdot \mathrm{Y})^2\right)} \tag{4}$$

On Screen Distance (OSD):

$$OSD = \mathrm{pixelsize} \times \sqrt{\left(\left(\mathrm{POG{\cdot}X} - \frac{\mathrm{x_{pixels}}}{2}\right)^2 + \left(\mathrm{y_{pixels}} - \mathrm{POG{\cdot}Y} + \frac{\mathrm{offset}}{\mathrm{pixelsize}}\right)^2\right)} \tag{5}$$

Angular accuracy (Ang_acc):

$$\text{Gaze angle } (\theta) = \tan^{-1}(\mathrm{OSD/dist}) \tag{6}$$

$$\mathrm{Ang\_acc} = (\mathrm{pixelsize} * \mathrm{Pix\_acc} * \cos(\mathrm{mean}(\theta))^2) / \mathrm{mean\_dist} \tag{7}$$

III. EYE GAZE ESTIMATION ALGORITHMS

Eye gaze tracking algorithms comprise of corneal reflection based methods which use NIR illumination to estimate the gaze direction or the point of gaze using polynomial functions, or a geometrical model of the human eye. 2D regression, 3D model, and Cross ratio based methods fall into this category. Another class of methods uses visible light and content information (e.g. local features, shape, texture of eye regions) to estimate gaze direction, e.g. appearance and shape based methods. The five different gaze tracking methods have their own advantages and disadvantages which are briefly discussed at the end of this section and summary of some key works on the development of these algorithms are presented in Table 2.

A. 2D REGRESSION BASED METHODS

In regression based methods, the vector between pupil center and corneal glint is mapped to corresponding gaze coordinates on the frontal screen using a polynomial transformation function. This mapping function can be stated as: $\boldsymbol{f}$: $(\boldsymbol{X_e}, \boldsymbol{Y_e}) \rightarrow (\boldsymbol{X_s}, \boldsymbol{Y_s})$ where X_e, Y_e & X_s, Y_s are equipment and screen coordinates respectively. The relation

TABLE 2. Classification of gaze estimation algorithms.

Method category	Paper reference	Setup (camera, LED)	Accuracy/ metrics (*deg= degree)	Tested for following operating conditions
2D Regression	[43]	Single steerable camera, multiple LEDs with 2 face camera in stereo	0.8 deg	Allows limited head movement
	[42]	LED with phototransistor	2.5 deg	No head movement
	[49]	1 video camera	1.4 deg,	No head movement
	[47]	4 LEDs and webcam	1.3 deg	Large head movement
	[45]	2 IR LED rings and 1 camera	5-8 deg	Significant head movement
	[48]	Two video cameras	1.5 deg	Natural head movement
	[46]	Two loops of infrared light sources	20 pixels	Head movement allowed
	[41]	Single camera, single infrared light source	0.87 deg	No head movement
	[44]	Webcam with built in LEDs	Nearly 2.5 deg	Limited head movement
	[51]	2 IR light sources, 1 webcam	1.11 deg	User distance
3D model	[56]	Pan tilt camera –stereo with LED arrays	1.0 deg	Free head motion
	[53]	1 camera 2 LEDs	1-3 deg	Full head pose compensated
	[4]	1 camera 2 LEDs	1-3 deg	Full head pose compensation
	[146]	1 camera no LED	1-2 deg	No head pose variation
	[57]	4 camera 2 LEDs, mirrors	0.6 deg	Head pose compensated
	[58]	2 camera 2 LEDs	Less than 1-2 deg	Head pose compensated
	[147]	2 camera 2 LEDs	< 1-2 deg	Head pose compensated
	[148]	3D face model , 2 camera, no LED	1 deg	Head pose compensated
	[54]	2 camera, 1 light, mirror	3 deg	Head pose compensated
	[55]	2 ring of lights, 2 cameras	1.25 deg	Small head pose tolerance
	[63]	1 depth sensor with integrated LEDs	3.78 deg	Head movement, user distance
	[64]	1 depth sensor with integrated LEDS	5 deg	Head pose
Appearance based	[83]	Eye images	4.87 % accuracy	-----
	[149]	VGA camera	3.5 deg	No head movement
	[145]	Webcam	1- 2 deg	Slight head movement
	[5]	VGA camera	4 deg	Allows head movement
	[78]	Eye images	1.5 deg	Slight head movement
	[79]	Eye images	2.3 deg	No head movement
	[150]	1 camera, NIR illuminator	0.5 deg	----
	[91]	Webcam	2.5	----
	[81]	Model, camera	3.5 deg	Free head movement
	[151]	2 cameras	< 3 deg	Free head movement
	[73]	Webcam	Recognition rate: 97 %	----
	[76]	1 camera	Correct adaptation rate: 91%	Tolerant to head pose, occlusion, low image resolution, illumination change
	[82]	Image databases	> 96%	-----
	[84]	Image databases	Recognition rate 97%	-----
	[87]	Image databases	6.3 deg	Head pose, illumination variations
	[71]	Image databases	< 7 deg	Head pose variations, occlusion
	[80]	Commercial camera	94 %	------
Cross Ratio based	[65]	4 IR LEDs, 1 camera	0.3- 0.4 deg without / 1-2 with head movement	Head movement allowed
	[67]	Camera with 16 IR LEDs	0.9 deg	Free head movement
	[68]	1 camera, 7 LEDs	1.4 deg	Head movement allowed
	[152]	Camera, 8 IR LEDs	0.3-0.4 deg	Free head movement
	[6]	1 camera, 4 light sources	0.5 deg without / 3.5 deg with movement	Head movement allowed
	[66]	2 cameras. 5 LEDs	0.98 deg	Large head movement
	[69]	1 camera, 4 NIR LEDs	10.3 mm	None
Shape based methods	[88]	1 camera	85%	None
	[89]	Image frames	4.5 pixels	None

can be presented as described in [41] and [42]:

$$X_s = a_0 + \sum_{p=1}^{n} * \sum_{i=0}^{p} a_{(i,p)} X_e^{p-i} Y_e^{i} \quad (8)$$

$$Y_s = b_0 + \sum_{p=1}^{n} * \sum_{i=0}^{p} b_{(i,p)} X_e^{p-i} Y_e^{i} \quad (9)$$

Where n represents polynomial order, a_i & b_i are the coefficients. The polynomial is optimized through calibration in which a user is asked to gaze at certain fixed points on the frontal screen. The order and coefficients are then chosen to minimize mean squared difference (ε) between the estimated and actual screen coordinates (with known camera

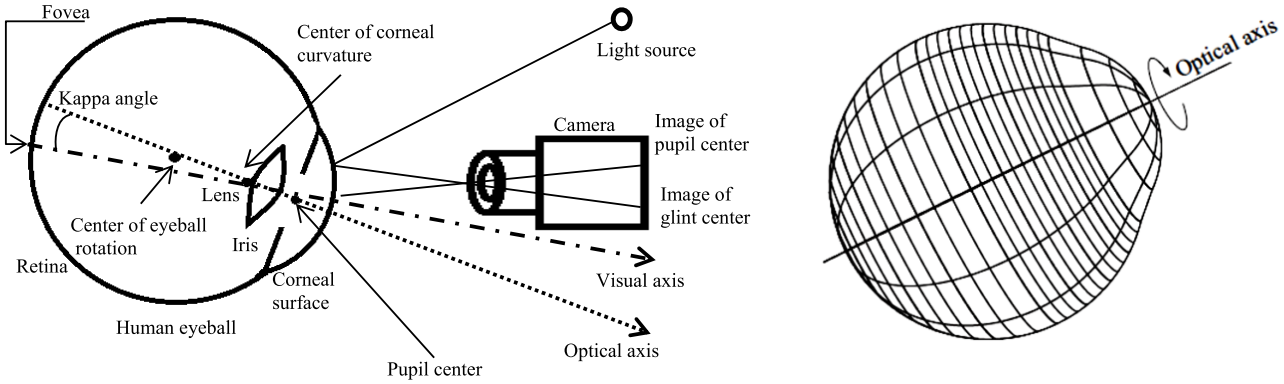


FIGURE 4. (a) Model of a human eye ball, eye parameters and setup elements used in 3D eye gaze tracking [32], [56]. The optical axis is shown as the line joining the center of curvature of the cornea with the pupil center. The visual axis passes through the fovea and the center of corneal curvature. Kappa angle is the angular deviation between the optical and visual axis. (b) An aspherical model of the cornea, as a surface of revolution about the optical axis of the eye [59].

coordinates of user gaze), which is stated as:

$$\varepsilon = (X_s - Ma)^T (X_s - Ma) + (Y_s - Mb)^T (Y_s - Mb) \tag{10}$$

Where a and b are the coefficient vectors and M is the transformation matrix given by:

$$a^T = [a_0 a_1 \dots a_m], \quad b^T = [b_0 b_1 \dots b_m] \tag{11}$$

$$M = \begin{bmatrix} 1 & X_{e1} & Y_{e1} & \cdots & X_{e1}^n & \cdots & X_{e1}^{n-i} Y_{e1}^i & \cdots & Y_{e1}^n \\ 1 & X_{e2} & Y_{e2} & \cdots & X_{e2}^n & \cdots & X_{e2}^{n-i} Y_{e2}^i & \cdots & Y_{e2}^n \\ \vdots & \vdots & \vdots & \cdots & \vdots & \cdots & \cdots & \cdots & \cdots \\ 1 & X_{eL} & Y_{eL} & \cdots & X_{eL}^n & \cdots & X_{eL}^{n-i} Y_{eL}^i & \cdots & Y_{eL}^n \end{bmatrix} \tag{12}$$

Where M is the transformation matrix, m the number of coefficients and L the number of calibration points [42]. The coefficients can be obtained by inverting the matrix M as [43]:

$$A = M^{-1} X_s, \quad b = M^{-1} Y_s \tag{13}$$

Cherif *et al.* [42] used a 5×5 point calibration routine and 2 higher order polynomial transformations while Cerrolaza *et al.* [44] studied effects of head movement on system accuracy with a 4×4 and 8×8 grid. Blignaut [41] also compared several mapping functions and calibration configurations with a 15×9 point grid and determined that number and arrangement of calibration targets and components of the mapping function play very important roles in determining overall accuracy of tracker. Robust and accurate gaze estimation under head movement was also achieved using neural networks by Zhu and Ji [45] and Jian-nan *et al.* [46].

Some other key works in this class of gaze estimation methods include Ma *et al.* [47] which introduces a 2D mapping algorithm that can handle unconstrained head movements and distorted corneal reflections due to various noise effects. It uses multiple geometrical transformation based mapping of CRs and demonstrates high reliability measures for different user distances and loss of CRs due to head/eye motion. A calibration free algorithm is detailed in Zhu *et al.* [48] using SVMs which is robust to natural head movement and achieves high accuracy (1.5 degrees) in presence of head movement Cerrolaza *et al.* [44] provided an exhaustive and detailed review of mapping equations and their impact on gaze tracking system response. The paper reports 400000 calibration functions, mapping orders and features to compare their impact on accuracy measures. Another important work is by Zhu and Yang [49] which presents a very high resolution gaze estimation method resistant to head pose without requiring geometrical models. In [50] an improved three layer artificial neural network is used to estimate the mapping function between gaze coordinates and the pupil-glint vector. This method is shown to achieve better accuracy than simple regression based methods. A new approach involving only 2 light sources instead of four is implemented in [51] where two IR LED induced glints are real and the other two are virtual ones computed mathematically. This is done to make the algorithm more suitable for consumer applications and simplify hardware. The PoG is then estimated using mapping function to relate the four glint locations and screen through a calibration process. A modified PCCR method is implemented in [52] to have improved tracking accuracy and suitable for indoor and outdoor use. In this, adaptive exposure control is proposed since the IR LED brightness variations within a PCCR based setup affects pupil detection and gaze tracking to a large extent.

B. 3D MODEL BASED METHODS

These methods use a geometrical model of the human eye to estimate the center of the cornea, optical and visual axes of the eye (Fig. 4a) and estimate the gaze coordinates as points of intersection where the visual axes meets the scene. 3D model based methods can be categorized on the basis of whether they use single or multiple cameras and type of user calibration required.

3D model based methods using single camera have been reported by Meyer *et al.* [53], Guestrin and Eizenman [32] and Hennessey *et al.* [54]. Single camera systems have simple

system geometry, no moving parts and fast re-acquisition capabilities. For 3D gaze estimation in Meyer *et al.* [53] a single camera and LED are used to achieve an accuracy of 0.5 degrees with user calibration. Guestrin presents a mathematical model to reconstruct the optical and visual axes of the user's eyes from the centres of the pupil and glint in the captured video frames and configuration of a remote gaze tracking system. The model considers single and multiple cameras and light sources in estimation of the point of gaze. It then demonstrates the gaze tracking performance of a system implemented using two NIR light sources and one camera using the model .Their method achieves an accuracy of around 0.9 degrees. The system proposed by Hennessey includes a single camera and multiple LEDs to achieve 3D gaze tracking with free head motion.

Multi camera methods achieve high accuracy and robustness against head movement but require elaborate system calibration procedures including calibration of cameras for 3D measurements, estimating positioning of LEDs and determining the geometric properties of the monitors and their relation with the cameras. Some key works using two or more cameras include Lai *et al.* [55], Ohno and Mukawa [56], Beymer and Flickner [57] and Zhu and Ji [37]. Ohno describes 3D gaze tracking allowing free head motion using simple two point calibration and a two camera system comprising of an eye positioning unit and a gaze detection unit. The eye positioning unit uses narrow field stereo cameras and controls direction of the gaze positioning unit to achieve head motion independent tracking.

A head pose free gaze tracking system is also implemented by Beymer with a wide angle stereo system for eye positioning and narrow angle stereo system for gaze detection. Pan and tilt directions of the narrow angle camera are controlled using rotating mirrors with galvo-motors. Shih and Liu [58] used a simplified eye model by Le Grand with two cameras and two LEDs to estimate the optical axis of the eye through solving linear equations. It uses single point calibration. The method in Zhu uses a gaze mapping function along with a dynamic head compensation model to update the gaze mapping function whenever the head moves to achieve tracking under natural head movement. It uses a 2 camera system with one time user calibration. With respect to the model of human eye, [32] provided evidence that a fully spherical corneal model will result in no impact of head movements on gaze estimation. They assumed an ellipsoidal model of the cornea and reported that gaze estimation errors increase with corneal asphericity and this also results in sensitivity of gaze estimation to head movements. An aspherical model of the cornea [59] is shown in Fig 4b and it is a surface of revolution about the optical axis of the eye. Use of this model showed to result in better accuracy, especially near the display corners as compared to a traditional 3D model based method.

Calibration free gaze estimation techniques have been proposed by Nagamatsu *et al.* [60], Model and Eizenman [61] and Morimoto *et al.* [4]. In [60], a calibration-free method is proposed using two pairs of stereo cameras, light sources and a spherical model of the cornea. One pair of cameras and two light sources are used for each eye to estimate the eye optical axis and the position of the center of the cornea. Optical axes of both eyes are measured using a binocular 3D eye model to estimate the point of gaze, achieving an accuracy of around 2.0°. Model & Eizenman also proposed a multiple camera based system to capture stereo images of the eye with corneal reflections. From the stereo eye images, eye features, such as the center of the pupil and corneal reflections are used to estimate subject-specific eye parameters. These parameters are then used with the eye features to estimate PoG. Their method is calibration free, has a tracking range of 3 to 5 meters and accuracy of less than 2 degrees. Morimoto proposed a method using two light sources and one camera that doesn't require user calibration for every session. It uses the Gullstrand model of the eye and ray tracing techniques to estimate the cornea and pupil centers. It achieves an accuracy of 2-4 degrees of visual angle dependent on the position of the light sources.

A new class of 3D gaze tracking has recently emerged with the usage of depth sensors in several works. These sensors comprise of an RGB camera and an infra-red depth camera. Typically resolution for the RGB camera is 640 × 480 pixels, with 45 degrees vertical and 58 degrees horizontal field of view. The depth camera resolution is about 1.5 mm at 50 cm. Gaze tracking using the consumer grade depth sensor (Kinect) is proposed in [62]. The method uses an eye model; 3D coordinates of eye features are obtained from Kinect and eye parameters like eyeball and pupil center are derived from a user calibration process. With this, 3D gaze coordinates are tracked in real time with a simple setup. Another work [63] reports the use of Kinect and a simple low cost setup for 3D model based gaze estimation allowing free head motion. It derives the 3D model parameters using convolution based means of gradients iris center localization method and uses a geometric constraints-based method to estimate the eyeball center. They assume that iris center points are distributed on a sphere originated from the eyeball center and the sizes of two eyeballs of a subject are identical. Kinect data is used to obtain 3D positions of person's head pose, iris and eyeball centers. Reference [64] also uses a Kinect sensor and a model to estimate eyeball center by making users look at a target in 3D space. Kinect is used to build a head model to determine the eyeball center, detect the pupil center and determine 3D eye gaze coordinates in conjunction with the eye model.

C. CROSS-RATIO BASED METHODS

These methods work by projecting a known rectangular pattern of NIR lights on the eye of the user and estimating the gaze position using invariant property of projective geometry. Four LEDs on four corners of a computer screen are used to produce glints on the surface of the cornea (Fig. 5). From the glint positions, the pupil and the size of the monitor screen, gaze location is estimated using two perspective projections. The first projection comprises of the virtual images of the

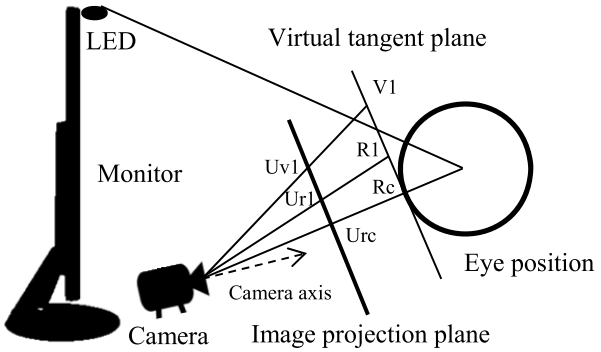


FIGURE 5. Setup for implementing cross ratio based gaze tracking [67]. Four light sources are used at four corners of the monitor screen (only one is shown here. V1 and R1 are the virtual projection and corneal reflection of L1, Rc is the reflection and the virtual projection of the LED fixed at the camera's optical axis. V1, R1 and Rc are projected to the image plane as Uv1, Ur1, and Urc.

corneal reflections of the LEDs (scene plane). The second projection is the camera projection, that is the images of the corneal reflections on the camera's imaging plane. With these two projections a single projective transformation relating the scene and camera image plane is obtained. Then the projection of the PoG on the scene plane to the image of the pupil center on the camera plane can be estimated [65].

Key works on the development and experimental verification of the cross ratio based methods can be found in Yoo and Chung [66] and Hansen *et al.* [6]. Coutinho et al. presents a detailed analysis of methods and comparison of their accuracy in [67] and [68]. They also suggest improvements by including a fifth LED on the optical axis of the camera and using a calibration procedure to improve accuracy [68]. Error compensation with polynomial-based regression have been proposed by Cerrolaza *et al.* [44] or Gaussian process regression [6]. Error correction by homography mapping that eliminates the need of the fifth light source has been proposed by Kang *et al.* [69].

D. APPEARANCE BASED METHODS

In appearance based methods the information from the eye region is represented using a model trained with a set of features extracted from eye images. In Bacivarov *et al.* [70], a statistical model is used to represent shape and texture variations and trained using images of the eye region annotated with landmark points (Fig.6). The shape vector is the concatenated coordinates of all landmark points, stated as

$$s = (x_1, x_2, \ldots, x_L, y_1, y_2, \ldots, y_L)^T \tag{14}$$

where L is the number of landmark points. The shape model is obtained by applying Principal Component Analysis (PCA) on the set of aligned shapes (equations derived from [70]):

$$s = \bar{S} + \varphi_s b_s,$$
$$\bar{S} = 1/N_s \sum_{i=1}^{N_s} s_i \tag{15}$$

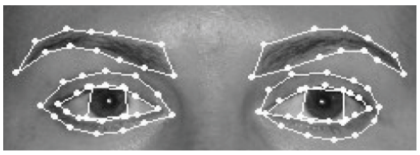


FIGURE 6. Image fitted with an Active Appearance model of the eye region [70].

where $\bar{S}$ is the mean shape vector, and N_s is the number of shape observations; φ s is the matrix having the eigenvectors as its columns; b_s is the set of shape parameters. Similarly, the texture vector defined for each training image is: $t=(t_1,t_2,\ldots,t_p)^T$ (p : number of texture samples). The texture model is derived by means of PCA on the texture vectors as (N_t: number of texture observations, $\bar{T}$: mean texture vector)

$$t = \bar{T} + \varphi_t b_t$$
$$\bar{T} = 1/N_t \sum_{i=1}^{N_t} t_i \tag{16}$$

The sets of shape and texture parameters (b_t) describe the appearance variability of the model:

$$c = \begin{pmatrix} W_s b_s \\ b_t \end{pmatrix} \tag{17}$$

(W_s is the vector of weights). This is the statistical model that an Active Appearance Model (AAM) algorithm uses to best fit the model to a new eye image.

An active appearance based method for retrieving eye gaze from low resolution videos is presented in [71]. Global and local appearance models are trained and fitted for the whole face as well for capturing variance of the face and eye regions. For classifying the eye gaze into six directions, two different approaches are adopted. Gaussian Mixture Models (GMMs) are trained for large changes in gaze angles and for small gaze changes a Histograms of Oriented Gradients based method are tested. A method for 3D gaze tracking without use of active illumination is proposed in [72]. In this a synthetic iris appearance fitting method is introduced that computes the 3D gaze direction from iris shape. The method synthesizes a set of iris appearances and then fits the best solution to the captured eye image. This is claimed to remove unreliable iris contour detection problems arising in simple ellipse fitting and requirement of high resolution images by other methods. Once the iris contour is accurately estimated, a 3D eyeball model is used to estimate gaze from the captured eye image using the iris center/shape information.

Several appearance based methods report use of local features with Support Vector Machine (SVM)s for classification of gaze direction. These include Wu *et al.* [73] in which an AAM is used to locate the eye region using 36 feature points that represent the contour of eyes, iris size, iris location, and position of pupils. Gaze direction is estimated from 2D coordinates of feature points and is classified using an SVM. In Lu *et al.* [74] Local-Binary-Pattern (LBP) is used

to calculate the texture features and a dual camera system is used to detect the space coordinates of the eyes. These two sets of information are fed into an SVM to classify the gaze direction under natural head movement. A novel method based on Local Binary Pattern Histogram (LBPH), is used in [75]. LBPH and PCA are used to extract eye appearance features and several classification methods based on SVM, neural networks and k-Nearest Neighbor (k-NN)s are tested for accuracy on a collected dataset for gaze estimation. The LBPH with SVM yields best accuracy. Chen and Liu [76] reports the use of a special kind of discriminatory Haar features and efficient SVMs (eSVMs) for implementing a computationally efficient gaze tracking method. Haar cascade is also used in [77] for real time gaze tracking. Rectangular features of the eye region are calculated to extract eye and pupil regions in an image which are mapped with the gaze coordinates on screen.

Neural network based approaches are used in [78] for head-pose tolerant gaze tracking. Training data comprised of cropped eye images of a user gazing at a given point on a computer screen and corresponding coordinates of that point. Neural networks are used along with a skin color model to detect the face and eye regions in [79]. An improved artificial neural network optimized using the Particle Swarm Optimization approach is used for fast, high accurate and robust gaze estimation with low resolution eye images in [80].

Some methods include use of 3D face models as in Lai *et al.* [81] in which head pose free gaze estimation is implemented using a such a face model with head and eye coordinate systems. Both eye appearance and head pose are considered as components of a high dimensional head pose and eye appearance (HPEA) space. A Neighbourhood Approximation Forests (NAF) approach is used to model the neighbour structure of the HPEA space followed by Adaptive Linear Regression to estimate gaze direction. Other approaches report use of a deformable model [76] and Genetic algorithms [83].

In recent times, deep learning (DL) and convolutional neural network (CNN) based methods have been proposed for gaze estimation. In [84], a three stage CNN model is used to classify seven gaze directions from images taken with low cost webcams without need for calibration. Gaze tracking for a near eye display robust to illumination, skin and eye color variations and occlusion is implemented using CNNs in [85]. The CNN is used to learn the mapping from eye images to gaze position and comprises of two convolutional layers and two pooling with a fully connected layer at the end.

In [86] a smartphone based app is created to collect eye images from 1450 participants which is used to train a CNN based gaze tracker that can run in real time and without calibration. The dataset comprises of images with widely varying backgrounds, lighting and head motion and the network is trained with crops of both eyes and the face region.

In [87] an extensive eye gaze dataset is built and a multimodal CNN based method is tested. The dataset contains more than 200,000 images with variable illumination levels and eye appearances. The CNN uses a convolutional layer followed by a max-pooling layer and second convolution layer followed by a max-pooling layer, finally with a fully connected layer. The CNN learns the mapping between input parameters, i.e., 2D head angle, eye image and gaze angle (output).

E. SHAPE BASED METHODS

These methods employ deformable templates of the eye region, using two parabolas for eye contours (Fig.7) and circle for the iris, and fitting them to an eye image [88]–[91]. The procedure is to find the similarity between the template of a chosen region with images of that region.

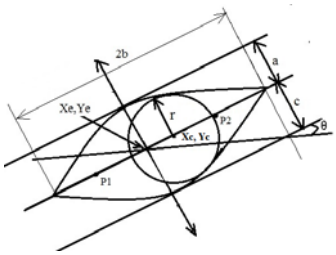


FIGURE 7. Template of an eye -region [88]. X_c, Y_c & X_e, Y_e represent the center of the pupil and the eye respectively. P_1 and P_2 are foci of two parabolic sections and a, b, c and θ their parameters, r is the radius of the pupil.

This can be done by normalized cross-correlation, modified cross-correlation or by mean square error calculation [89]. If the template of pixel intensities of a region is represented by T(u,v) and I(i,j) represents that of the captured image then S(i,j) is the similarity measure between the template and image. If cross correlation is used as similarity measure, then S is given by:

$$S(i,j) = \frac{< T \times I_T > - < T >< I_T >}{\sigma(T)\sigma(I_T)} (18) \qquad (18)$$

where <> is the average operator and $< x >$ is the pixel-by -pixel product given by:

$$< T > = \frac{1}{n} \sum_{u,v} T(u,v)$$

$$< T \times I_T > = \frac{1}{n} \sum_{u,v} T(u,v) I(i+u, j+v) \qquad (19)$$

σ is the standard deviation of the area being matched.

$$\sigma^2(T) = \frac{1}{n-1} \sum_{u,v} (T(u,v))^2 - < T >^2 \qquad (20)$$

The mean squared error similarity measure is given by:

$$S(i,j) = \frac{1}{n} \sum_{u,v} (T(u,v) - I(i+u, j+v))^2 \qquad (21)$$

F. SUMMARY AND DISCUSSIONS

The different gaze estimation algorithms presented above have distinct characteristics, advantages and disadvantages.

The 2D regression based methods utilize the features of the human eye, like eye geometry, pupil contours and corneal reflections and can be implemented using a single camera and a few NIR LEDs. However, these techniques are very vulnerable to head movements and require users to hold their head very still using a head rest, chin rest or bite bar.

3D model based methods have tolerance towards user head movement and most of them allow free head motion. However the hardware requirements for implementing 3D and stereo gaze tracking methods are high as they need several light sources or multiple cameras.

Cross ratio based methods have advantages, e.g. they do not need an eye model or hardware calibration and allow free head motion. But they are affected by problems such as increased error with distance of user and user dependent factors.

Appearance model-based algorithms are non-PCCR methods that use the shape and texture properties of the eyes and position of the pupils relative to the eye corners to estimate gaze. These methods have low hardware requirements which make them suitable for implementation on platforms without a high-resolution camera or additional light sources. The disadvantage is that their accuracy is mostly lower than PCCR based methods that degrade with head movements, variation in illumination levels and for robust performance they need large training image databases.

Shape based methods have been implemented for 2D gaze estimation with low-resolution webcam images achieving accuracy around 2°. However, downside of these methods include problems due to head pose variations and eye occlusions, adapting to largely variable eye shapes, computational complexity and issues with model initialization.

In Table 2, some key research works on the above gaze tracking algorithms are presented. Column II presents the reference to the individual papers; Columns III to V presents various characteristics of the methods reported in them. The table highlights basic features and differences among different gaze tracking algorithms. 2D regression and appearance based methods have simple setups but typically offer accuracy values around 2 or 3 degrees. The accuracy of 2D regression based methods can be improved by increasing number of calibration points and using a chin rest to obtain fixed head position. On the other hand, 3D and cross ratio methods require more elaborate setups but offer much better accuracy (around 0.5 degrees) and most of them allow head movement.

IV. USER PLATFORMS IMPLEMENTING GAZE TRACKING

In this section, the user platforms where eye gaze tracking has been implemented are described and classified.

A. DESKTOP BASED SYSTEMS

Applications of eye gaze on desktop systems fall into several categories, such as computer communication, password entry and psychoanalysis. Sibert and Jacob [35] developed gaze based fast object selection as a substitute for mouse. A gaze based application called MAGIC (Manual And Gaze Input Cascaded) is presented by Zhai *et al.* [92] in which gaze based pointing is reported to have higher speed and accuracy than manual pointing. In Ghani *et al.* [93], a Hough transform based pupil detection for gaze based control of a mouse pointer is proposed. In Agustin *et al.* [94] evaluation studies on the use of eye gaze in video gaming control for target acquisition and tracking are made. Gaze input had a similar performance to the mouse and joysticks for big targets.

Kasprowski and K. Harężlak [95] and Kumar *et al.* [96] reported the use of gaze to enter a password using gaze tracking. A series of user fixations on specified digits formed the password sequence. Methods robust to shoulder surfing problem were reported by Bulling *et al.* [97] where a computational model of visual attention is used to increase security. Applications of using eye gaze patterns for identifying user tasks, mental workload and inferring context of events and user distraction were reported by Iqbal and Bailey [98] and Doshi and Trivedi [99].

B. TV AND LARGE DISPLAY PANELS

There are recent applications of long range gaze estimation that use corneal reflection (CR) techniques for tracking gaze on large displays and smart TVs. Gaze movements can be used to select and navigate menus, modify display properties, switch channels and understand user interests. In Gwon *et al.* [7] a robust pupil detection method for gaze tracking on large display is presented using a wide and a narrow view camera with Adaboost and CAMShift algorithms. In another work, Lee *et al.* [8] reports a system for gaze tracking on a large-screen 60 inch TV based on a 2D method with geometric transform, using pupil center and four corneal specular reflections.

C. HEAD-MOUNTED SETUPS

Head mounted gaze trackers are portable platforms with applications ranging from computer input, interactions in virtual environments, gaming controls, augmented reality and neuro/psychological research. The general setup includes two cameras; one (eye camera) pointed at the wearer's eye, to detect the pupil; and the other (scene camera) capturing the wearer's point of view, with sometimes additional components like NIR light sources and hot mirrors. Head-mounted gaze trackers have been implemented as attachment-free, mobile, low-cost, lightweight devices with simple hardware and software. Also they are known to provide high accuracy gaze information in unconstrained settings.

3D gaze estimation with head-mounted trackers have been reported in several papers including [9], [100]–[104]. Algorithms for high accuracy and 3D eye tracking proposed by Lee *et al.* [100] are based on 3D human eye model

and Purkinje images. Pupil size and Purkinje images are fed as inputs to a multi-layer perceptron to estimate the depth location of gaze followed by 2D gaze coordinates. Takamatsu *et al.* [104] describes a Visual SLAM technique to estimate the user head pose and determine 3D point of regard of the user. It also includes a 3D environment to detect objects of focus and visualization of a 3D attention map in a real environment. Lanata *et al.* [9] presents a stereo-vision based method that implements a novel binocular system comprising of two mapping functions: linear and quadratic for depth estimation of gaze locations in 3D space.

The performance compared to other 3D methods show significant improvement in accuracy. In [103], 3D gaze tracking with multiple calibration planes is implemented with a single eye view camera and four IR markers. The system operates in monocular and binocular mode independently. A low-cost 3D eye tracking solution is provided by [101] in which images from two eye cameras are used with intensity thresholding and blob-contour detection to determine pupil center coordinates. The 2D gaze location is then obtained by polynomial mapping and 3D gaze is obtained from the meeting point of the gaze vectors of both eyes.

2D methods and applications with head-mounted eye tracking have been proposed in a several works. Reference [10] reports an ultra-low cost 2 camera based gaze tracking system that is easy to assemble and aligns scene and eye videos by using synchronized flashing lights. A novel smooth-pursuit-based calibration methodology is proposed that works with pupil detection to track gaze. A new high speed binocular eye localization scheme using simple components was proposed by Long *et al.* [105] using two cameras and two hot mirrors. It shows speed advancement over contemporary slow systems that do offline processing. Other advanced methods are proposed in Schneider *et al.* [106] where a head mounted camera system tracks and follows the user's gaze direction for natural exploration of a visual scene by capturing the perspective of the mobile user. Applications of this system include documentation of medical processes, sports etc. In Virtual and Augmented reality (VR and AR respectively) research, gaze tracking and gaze based functions are used to make the user experiences more immersive, natural and user interactions fast and efficient in a VR/AR environment. Applications for Augmented Reality are described in Lee *et al.* [107] where the device uses a scene camera and eye tracker to estimate user gaze and blink-state for interacting with the AR environment. A head mounted system that employs gaze tracking for immersive and realistic gaming experience is proposed by Lee *et al.* [108] where cursor aiming in the gaming environment is controlled by gaze tracking. A wearable gaze tracking device for determining effectiveness of text layout and line spacing for analyzing reading behavior is described in Toyama *et al.* [109]. Li *et al.* [11] presents the development of a head mounted system called OpenEyes with open hardware and software tools for gaze tracking. Several types of eye and head gestures, such as saccades, smooth pursuit and nod-roll are studied in [110] as interaction methods in a head mounted VR device. A head mounted display coupled with an eye tracker is used to study which of these gestures result in better user experience in a VR environment.

A facial re-enactment method for gaze aware VR system is presented in [111]. It does real time facial motion capture of a user wearing an HMD along with monocular eye tracking. The purpose is to achieve real time photo-realistic rendering of face and eye appearances as users change facial expressions and gaze directions. Gaze is estimated from images taken using monocular camera and IR LEDs using a hierarchical classification method. In [112] eye tracking is used to develop an immersive 3D user interface for VR and implement several multimedia applications along with usability testing. Eye gaze pattern is extracted along with fixations to implement 3D virtual menu selection and eye based typing for mail composition using a virtual keyboard. A saliency based gaze localization approach using deep learning has been proposed in [113]. In this, image features and head movements are inputs to a convolutional neural network to estimate gaze coordinates in a VR system.

To make eye tracking in HMD systems flexible with respect to various users, HMD drifts and tracker camera adjustments, [114] presents a method for automatic calibration of the eye camera in the HMD. With this, the users need not maintain a fixed head pose and relative movements of HMD and eye camera can be tolerated for reliable eye tracking. Reference [115] makes use of gaze, dwell time and half blink information for the purpose of hands-free object selection with an optical see through head mounted AR device. The multiple input parameters are used to avoid accidental/unintentional selection of items. Pupil motion is tracked to estimate a user's viewing point, with dwell time and half blink detection successively used for the object selection process. Kalman filtering is used minimize pupil jitter and drifts. The gaze tracking accuracy of the system is reported to improve with more calibration points with the best being 0.39 degrees.

An innovative AR application combining eye tracking with a smartwatch for implementing a wearable context aware messaging service is done in [116]. A head-mounted eye tracker estimates the gaze and together with data from a scene camera it is used to track where the user is looking in the real world. The smartwatch works as a message input and output device in conjunction with the tracking device to embed/display messages with the ''augmented objects. An AR application for reading and document retrieval using a head-mounted eye tracker along with a see through display is shown in [109]. Eye tracking helps to estimate the section of the document the reader is focused into, in real time. Associated information on the specific part of the document is retrieved and displayed on the HMD. A calibration procedure for the HMD is also mentioned. In this, a user is presented with several dots in the HMD and he/she has to click the position of each dot in a calibration window. The homography between the scene image and the HMD is

then estimated to map the gaze position on the scene to that on the HMD. The system achieves accuracy sufficient to distinguish between alternate lines in a document. A hardware and software framework for AR displays implementing eye tracking and gaze based interactions can be found in [117]. In this, a see through HMD is integrated with stereo eye tracking and configurable optics for various see through configurations. The system is designed to have on demand zoom and FOV expansion in a see through AR system. Eye related parameters like gaze, squint and blinks are tracked for activating different functions such as binary, sub-regional and gradual zoom and capturing snapshots of the AR view.

D. AUTOMOTIVE

Visual features of the face and eye regions of an automobile driver provide cues about their degree of alertness, perception and vehicle control. Knowledge about driver cognitive state helps to predict if the driver intends to change lanes or is aware about obstacles and thereby avoid fatal accidents. Gaze related cues that indicate driver attentiveness include: blink rate, temporal gaze variation, speed of eyelid movements and degree of eye openness. Several works for driver assistance systems have been reported that are based on video based gaze tracking using a variety of classifiers.

SVM based gaze classification is common on automotive platforms and has been reported in Liu *et al.* [118], Lee *et al.* [119] and Chuang *et al.* [120]. A real time gaze tracking method robust to variable illumination levels and driver wearing glasses in an automotive environment is reported in [118]. It uses Kalman filtering and mean-shift method to track driver eyes based on their locations in a previous frame. SVM with linear, polynomial and Gaussian kernels are used for eye verification. Lee *et al.* [119] proposed a robust SVM based driver gaze zone estimation system that works during day and night conditions and is robust to driver wearing eyeglasses. A multiclass SVM constructed from multiple two class (binary) SVMs was constructed to classify 18 driver gaze zones using a large database of 18000 gaze feature vectors. Another SVM based gaze zone classifier that takes face parts, i.e., mouth, eyes, and nose locations is reported in Chuang *et al.* [120]. A multiclass linear SVM is used which inputs the feature descriptor to output 8 gaze directions. Use of other type of classifiers is reported in Tawari and Trivedi [121] and Oh and Kwak [122]. In [121] a distributed camera setup is used for estimating gaze zones combined with head pose dynamics. A random forest classifier is used with static and dynamic features (head pose angles and time series statistics) to classify 8 gaze zones for the driver. In [122] Viola Jones method is used to detect the face followed by using linear discriminant analysis (LDA) to extract features from the eye region for classification. k-nearest neighbour method with Euclidean norm is used to classify the obtained features into seven gaze directions.

Gaze estimation in automotive using Purkinje images and PCCR methods are reported in Ji and Yang [17], Salvucci and Liu [123], Batista [14], Choi and Kim [124] and Ji *et al.* [125]. Several papers report the application of gaze information along with other facial and visual parameters for derivation of driver psychological/cognitive state. Key works include Bergasa and Nuevo [126] which reports a method for driver vigilance estimation using fuzzy classifier taking six parameters: Percent eye closure (PERCLOS), eye closure duration, blink frequency, nodding frequency, face position, and fixed gaze. A dynamic Bayesian Neural network approach is used in Sung *et al.* [127] for driver fatigue detection using gaze location along with face detection, eye positioning and iris tracking. Ji *et al.* [125] combines features such as eyelid and head movement, gaze, and facial expressions with a Bayesian network to determine driver fatigue. Another important work that considers eye movements for deriving driver cognitive state is [16]. It takes into consideration eye movement features like fixations, saccades, and smooth pursuits and calculates 16 different eye movement features. These are then fed to an SVM, static and dynamic Bayesian networks to do their performance comparison for prediction of driver distraction state. Reference [128] presents a low cost system to detect eyes-off-the road condition of a driver using facial feature tracking, 3D head pose and gaze estimation. A monocular camera is installed close to the steering wheel for tracking a driver's facial landmarks and accurate estimation of driver pose, location and gaze direction. Then, with 3D analysis of car/driver geometry the driver's eyes off the road condition is predicted in real time. In [129], driver gaze behaviour is studied to evaluate driving performance of a user when they had to interact with a portable navigation system while driving. Glance frequency and glance time were estimated to study impact of varying display sizes and positions of the navigation device while in use during driving. A portable driver assistance system involving driver drowsiness detection and eye gaze tracking is implemented using a Raspberry Pi and machine vision algorithms in [130]. A new multi-depth calibration approach is presented in [131] for obtaining 3D user PoG with stereo face cameras and monocular scene cameras for performing driver intent and actions prediction.

Several works study the dynamics between head pose and gaze behavior of drivers. In [132] the relationship between gaze and head pose are studied and regression models are developed to predict gaze location from the position and orientation of a driver's head. In [133] head and eye poses of 40 driving participants are estimated from monocular video and relative significance of head vs eye movements in gaze classification is studied. It is observed that driver behavior can be grouped into two cases, i.e., when the head moves a lot and gaze classification is mostly affected by head pose. The other case is when the head stays still and only the eyes move and thus accurate classification requires study of eye pose.

E. HAND-HELD DEVICES

Smartphones and tablets provide a unique paradigm for gaze tracking applications. Gaze tracking on handheld

FIGURE 8. Driver monitoring cameras mounted on car dashboard [124].

FIGURE 9. Eye tracking implemented in near real-time on a tablet. The tracking algorithm uses cascade classifiers and shape-based approaches to determine the eye region and centers. Elliptical model-fitting and 3D back-projections are then used to determine the eye optical axes and point-of-gaze [18].

devices is done using the device front camera, one or more IR light sources and various computer vision algorithms. Key approaches for handheld-eye tracking are reported by Vaitukaitis and Bulling [134] for tracking 3D gaze location from a video using a face and eye detector followed by edge detection and ellipse fitting to determine the eye limbus boundaries. In Lukander [135] a commercial eye tracker is integrated with a magnetic position tracking device to track positions of head and eyes. Holland *et al.* [136] uses a single perceptron neural network that maps the features of the eye region to a position on the screen. In Imabuchi *et al.* [137] eye tracking on a tablet is realized with blob and contour detection for deriving iris contours. Then center and planar homography transform is used to convert the coordinate of iris center to display coordinates.

Several works use the Haar classifiers/Viola-Jones technique [138] such as Elleuch *et al.* [139] which describes a system for android tablets to track head motions and eye gaze gestures from video captured using device front camera. The system is claimed to be robust to user movement and lighting conditions. In Pino and Kavasidis [140] eye tracking for tablets is implemented using a Haar classifier based eye detection module in conjunction with the CAMSHIFT algorithm.

Potential applications of eye gaze in handheld devices are presented in Nagamatsu *et al.* [19] describing a user-interface called Mobigaze that uses gaze to operate a handheld device. The system combines gaze and touch for operating the device interface thereby eliminating the Midas touch problem [31]. The Midas touch effect [31] refers to the phenomenon in which an eye-gaze-sensitive user interface cannot distinguish between user glances for collecting visual information and those for command input. Thus every user fixation may lead to activation even without the actual user intention. Another multimodal gaze based interaction method is presented by Drewes and Schmidt [141] for monitoring applications using dwell-time and gaze gesture. The EyePhone introduced by Miluzzo *et al.* [142] controls the phone functions with only gaze and blinks and is free of user touch. It describes an application (Eyemenu) in which gaze direction is used to selectively access and highlight menu buttons. A gaze based user authentication scheme for smartphones is implemented in Liu *et al.* [143] by making the user eyes to follow a moving target on the device screen. In Sun *et al.* [144], a 3D image display is combined with gaze estimation to achieve adaptive 3D display on a mobile phone that shows different views to corresponding viewing angles. In Imabuchi *et al.* [137] a gaze based communication interface is implemented for gaze based keyboard input and gaming.

Table 3 presents key information obtained from selected papers based on five gaze user platforms. It can be seen that different research works present their results in widely different formats and describe performance of their system under varying operating conditions.

F. USER PLATFORMS FOR GAZE TRACKING: SUMMARY OF USER CONFIGURATIONS

The user conditions while they are using gaze tracking in the above five platforms are completely different. Similarly, the operating environment and tracking setup are also unique to each platform. Table 4 presents typical system and user configurations for the five gaze tracking platforms including users' postures and viewing angles, screen sizes, typical distance between the user and the screen-camera setup. It can be seen that gaze tracking on different user platforms encounter wide range of operating conditions and therefore have diverse performance measures.

Apart from the differences seen in Table 4 above, Table 3 also shows the inconsistency in performance reporting formats among the different user platforms. Accuracies are reported in varied units –such as degrees, pixels, percentage of correct detection etc. Another feature as seen from Column 5 of this table is that only a few papers study the impact of operating conditions on the system performance. Further discussions about these aspects of gaze research are made in Section V.

Amongst the papers reporting in degree measures, a trend is seen in the accuracies for different eye gaze user platforms. Typically, it is seen that head mounted systems report better tracking accuracies of less than one degree amongst other platforms. For desktop systems it varies from 0.5 to 2 degrees of angular resolution and above 2 degrees for more dynamic platforms like automotive and handheld devices. Further, all platforms have unique setup requirements and users may

TABLE 3. Summary of research on eye gaze use cases in various platforms.

Use case/ Platform	Paper reference	Setup (camera, LED)	Accuracy/ metrics (*deg= degree)	Tested for following operating conditions
Desktop	[97]	2 integrated infrared cameras	Euclidean distances, 21 and 46 pixels	Distance from tracker
	[95]	Commercial tracker, 1 camera	61.1%	User dependent
	[96]	Commercial tracker, 1 camera	Error rate 15%	None
	[92]	Commercial tracker, 1 camera	Completion time, no. of hits/ misses	None
	[93]	1 webcam	Recognition rate 100%	None
	[141]	Commercial eye-tracker, 1 camera	Percentage, time	None
	[94]	One camera	Mean error rate 22.5%	None
	[98]	Commercial tracker, 2 eye/1 scene cam	Percentage time spent	None
	[153]	1 eye camera one scene camera	Number of fixations, time of fixations	None
	[99]	2 cameras	Gaze shifts/ not mentioned	Head motion
TV	[8]	1 camera, zoom-lens, 4 NIR illuminators	60.4 pixel, 1.32 deg	None
	[7]	1 wide & 1 narrow view camera	5.75 pixels	Occlusion
Headmounted	[100]	Eye-capture camera and NIR (LED), zoom lens	0.961 deg 1.601deg, (X, Y axis)	Distances from the screen
	[9]	Wireless camera, no LED	0.85 deg	Head motion, light level changes
	[10]	1 eye camera, 1 scene camera, 1 LED lamp	0.752 deg	None
	[11]	Eye camera, scene camera, LED	0.60, 1.03, 1.04 deg	Field of view
	[102]	2 eye tracker cameras, 1 head camera, hot mirrors, 2 LED	0.54 deg	Head motion, user motion
	[104]	2 eye, 1 scene camera, 1 IR LED	3.38 ± 2.38 deg	Head pose
	[105]	2 camera, 2 hot mirrors, IR LED	less than 0.5 deg	Occlusion
	[107]	1 eye camera, 1 scene camera	17.6 pixels, 0.81 deg	Eye limits, midas touch
	[109]	2 eye cameras, 1 scene camera. 6 IR LEDs	0.5 deg	Distance from tracker, viewing angle
	[103]	3 mini-cameras and 2 hot-mirrors	1.2 deg	Distance from tracker
	[12]	2 cameras	< 2 deg	None
	[154]	1 eye camera, 1 scene camera, 1 LED	0.66 deg	Distance from tracker
	[21]	2 eye , 1 scene camera	0.25 deg	Head pose
	[155]	2 Cameras	0.5 -3.5 deg	None
	[156]	Eye camera, scene camera, mirror	0.66deg	None

TABLE 3. *Continued.* **Summary of research on eye gaze use cases in various platforms.**

Use case/ Platform	Paper reference	Setup (camera, LED)	Accuracy/ Metrics (*deg= degree)	Tested for following operating conditions
Automotive	[13]	2 cameras	5 deg	Head pose
	[17]	2 cameras, 2 rings of LEDs	98-100 %	None
	[157]	Commercial tracker -2+2 cameras	3 deg	Head pose
	[124]	1 Camera	Not stated	Head pose
	[158]	2 cameras/ commercial tracker	0.18 radians	None
	[159]	1 camera	Not mentioned	Head motion
	[15]	2 cameras- 1 facing driver, 1 looking out	Accuracy 94.9%	Head pose
	[127]	1 camera	Detection ratio 85%.	Spectacles
	[125]	1 camera, 2 rings of LEDs	Classification accuracy 97.5%	Head pose
	[126]	IR camera on dashboard, 2 IR LED rings	Correctness rate 90-100%	Head pose, glasses
	[119]	2 NIR illuminators, a camera	Success detection rate 97%	Head pose/face pose
	[160]	1 camera	Detection rate 86-100%	Head pose
	[14]	NIR LED rings,1camera	Not mentioned	Face pose
	[161]	4 motion capture + 1 video camera	10 pixels	None
	[16]	2 cameras	5 deg	Head pose
	[123]	Head-mounted eye tracker	1 deg	None
	[162]	1 camera	Not mentioned	Head pose, lighting
Handheld devices	[134]	Device front camera	67.3% on a laptop	None
	[140]	Front camera of iPhone (1 camera)	79.1 % accuracy	Hand motion
	[18]	1 front camera	6:88 deg	None
	[136].	1 tablet front camera	3.47deg	None
	[137]	1 front camera	0.77 deg	None
	[144]	1 device front camera	Average pixel error	Head positions
	[108]	1 camera	Pixels	Head motion
	[142]	1 camera	Percent accuracy 70-98%	Accuracy with screen positions
	[143]	1 camera	True positive rate	None
	[19]	2 cameras with IR-LED	Not discussed	
	[139]	1 camera	Detection rate 60%	None
	[163]	1 camera	Detection accuracy 80-100%	Distance from user
	[135]	Commercial tracker, 2 eye 1 scene camera	0.15 Deg RMS	None
	[164]	1 front camera	Recognition rate 93%	Distance

assume various physical poses as seen in Table 4. Therefore, a general eye tracker may produce significantly different results depending on platforms.

TABLE 4. Features of gaze estimation systems in different user platforms.

User platforms	Distance	Viewing angle (degree)	Screen size (inch)	User condition
Desktop	30-50 cm	~40	14-17	Upright, sitting, static
TV panels	2-5 m	40-60	26-70	Lean back, sitting, static
Head mounted	2-5 cm	55-75	--	Lean independent, sitting or standing, static or mobile
Automotive	50 cm	40-60	--	Upright, sitting, mobile
Handheld devices	20-40 cm	5-12	5-10	Lean forward, sitting or standing, mobile

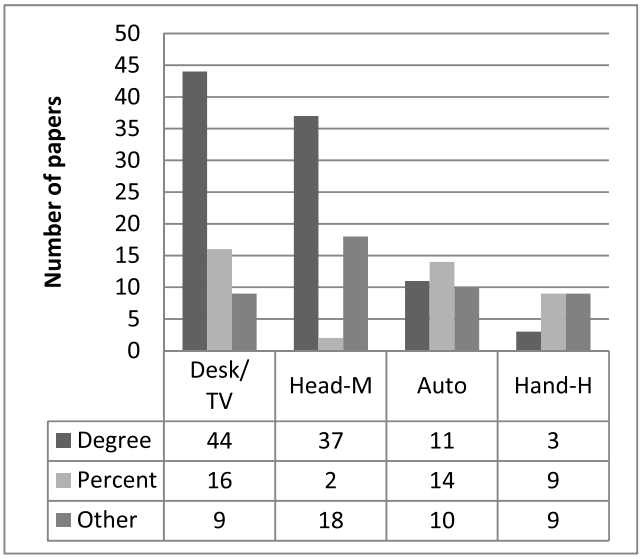


FIGURE 10. Diversity in metrics used for representing performance in different platforms (Head-M is head-mounted, Auto is for automotive systems and Hand-H represents handheld devices like smartphones and tablets that implement gaze tracking). The figure shows that although degree measures are most common format of measurement, results are reported in several other formats which cannot be linked to each other. The inhomogeneity is majorly observed in automotive and handheld platforms.

V. PERFORMANCE METRICS AND PLATFORM SPECIFIC ERROR SOURCES IN EYE GAZE RESEARCH

A. DIVERSITY OF GAZE ESTIMATION PERFORMANCE METRICS IN DIFFERENT USER PLATFORMS

In contemporary literature, research works on gaze tracking present the results of their algorithm or application in several ways. The common measures used are angular resolution (in degrees), gaze recognition rates (in percentage) and minimum pixel shifts/distance between gaze and target locations. These metrics are no way correlated to each other and each research work defines some metrics, for example percent recognition rates or error rates in their own way. The consequence is that most gaze estimation methods cannot be inter-compared.

To better understand the diversity of performance measures used in gaze research, an information statistics is collected from our reviewed papers and presented in Table 5. Firstly, the papers are grouped according to platforms and then the performance measures reported in each paper are classified according to the categories: degrees, percentage and ''others'' (pixel shifts/distance in mm). In Table 5, Column 2 has the total number of research papers that was considered in the survey for each platform. Columns 3, 4 & 5 provide reference to papers that reported their performance measures in either ''degrees'', ''percentage'', or ''other'' formats respectively. Fig. 10 presents this statistics i.e. number of papers that reported accuracies in various formats for four different user platforms.

What can be observed from Table 5 is that there is no standard convention that is used to represent performance scores of eye gaze estimation algorithms in contemporary eye gaze research. For example, as seen in the table, out of total 69 papers from desktop/TV platform, 44 papers report tracking accuracy in degrees while 16 papers report them as gaze recognition rates and 9 papers in mixed units. Also this discrepancy is found in literature for all user platforms as seen from the rows 3 to 5 of the table and Fig. 10. From Fig. 10, we observe that degree measures are common units used for desktop and head-mounted platforms, whereas the representation is completely heterogeneous for automotive or handheld devices. The result of this inhomogeneity is that, the stated performance from a large volume of research in this field can neither be compared nor interpreted quantitatively.

B. PLATFORM SPECIFIC FACTORS AFFECTING USABILITY OF GAZE TRACKING SYSTEMS

Eye gaze estimation systems on various user platforms and applications face a broad range of operating conditions that are rarely taken into consideration or characterized during their development. The practical performance of a gaze tracking system in reality may be affected by several factors (or we may call them: sources of errors) that are common or unique to each platform. Some of these factors and their effects have been discussed in our earlier work [24].

For example, the major sources of errors in desktop are head movements. In head-mounted trackers, errors may arise due to ''Midas-touch'' effect [31], miss-calibration or tracker latency. Errors in automotive systems may arise from platform and user head movements, variable illumination, occlusion due to shadows or user wearing glasses. In handheld devices, eye tracking gets highly challenged by changing positions of the user with respect to the device, head pose, hand jitter, variable illumination, and Midas-touch.

In gaze research, effects of some of these conditions such as head-pose changes, user distance and viewing angle are

TABLE 5. Diversity in performance metrics in gaze estimation systems.

Platform	No. of papers	References where following performance metrics are used		
		Degree (95 papers)	Percentage (41 papers)	Others (46 papers)
Desktop and TV panels	69	[5][65][67][68][152][6][66] [165][166][41] [44][42][47] [56][48][43][45][57][54][55] [4][53][148][58][147][167] [146][37][91][168][145][79] [150][8][169][51][52][62][63][64] [85][87][71][72]	[76] [83] [82][73][88] [141][93][95][96][170] [171][94][172] [84][75][80]	[74][89][97][173][92] [174][98][153][77]
Head mounted	57	[101][100][9][10][175] [176][177][178][102][106] [104][179][180][165][181] [182][105][107][183][109] [106][103][184][185][12] [154][186][187][156][188] [22][155][21][114][115][109][50]	[189] [117]	[11][176][190][191][192] [193][194][195][196][197] [198][35][199][200][156] [110] [111][112]
Automotive	35	[13][157][123][16][158][201] [202][131][128][129][132]	[203][162][15][17][122] [121][125][119][120][16] [127][160][204][133]	[14][205][206][159][124][161] [207][204][208][130]
Hand-held	21	[136][137][18]	[139][209][164][210][163] [143][142][140][134]	[211][212][141][108][213][135] [19][144][86]

studied but other factors such as display properties i.e. size and pixel resolution of the screen where gaze is tracked, platform movement and jitter, illumination changes, camera quality and human eye limitations are very sparsely reported. To study this scenario, Table 6 presents an information statistics on practical conditions that may affect gaze tracking and the extent to which they have been studied. Firstly the different error sources are listed in Column I. Then the papers are categorized according to platforms to identify the research works where these error sources have been reported. This information is presented in the different columns of Table 6.

It is seen from the table that each user platform encounters at least 5-6 different conditions which can affect their performance but their impacts are very sparsely studied for all user platforms. Head-pose is the most studied factor out of all but very few papers analyze the impact of others error sources.

Further in presence of these factors, the real performance of an eye tracker can deviate significantly and unpredictably from the scores reported under ideal conditions. Hence, unless these error sources are adequately evaluated, the accuracy achieved by a gaze tracking system cannot be reliably specified.

VI. A PERFORMANCE EVALUATION FRAMEWORK FOR EYE GAZE ESTIMATION SYSTEMS

A. NEED AND RATIONALE FOR DEVELOPING COMPREHENSIVE PERFORMANCE EVALUATION STRATEGIES FOR GAZE ESTIMATION SYSTEMS

From the sections and tables above it is seen that currently there exists a large diversity among gaze tracking methods, setup, implementation platforms and accuracy metrics.

An important issue that becomes apparent is that there is currently no comprehensive practice for realistic performance evaluation of gaze tracking systems. Most research works do not assess their systems under the impact of various error sources or provide adequate details about their system configurations. For example, Table 6 shows that only 35 out of 69 papers on desktop based systems and 16 out of 57 papers on head-mounted systems report effect of head pose variations. Only 2 works in each of desktop and head-mounted platforms report effect of display properties of the gaze tracking setup. Effect of illumination changes is reported in 4 papers in desktop and one in head-mounted systems. In dynamic platforms like automotive and hand-held devices, where external conditions are more variable, the evaluation statistics is even poorer, as seen from Table 6. Another issue observed is that the accuracy scores for gaze based systems are presented using varied formats and metrics, such as angular resolution, correct detection rate, pixel and physical distances etc. which makes them difficult to interpret and inter-compare. For example: Table 5 and Fig. 10 show that out of 182 total research works across all platforms, 95 papers report gaze accuracy scores in degrees whereas 41 papers report in percentages (correct detection rates) and 46 papers use other heterogeneous units.

There is also considerable ambiguity in terminologies used in eye gaze research. For example, mentioning ''slight head movement'' (as in [43] [78], and [145]) or ''large head movement'' (in [45], [47], and [66]) and free head movement (in [53], [56], and [67]) gives no quantitative idea about the real extent of head pose variations tolerated by these gaze tracking systems.

Considering these factors, the development of comprehensive evaluation strategies for gaze tracking systems seems necessary for several reasons: a. to study impact of various error sources on system performance b. to report system performance quantitatively in uniform formats c. compare results from different eye tracking systems under different operating conditions d. to identify the main bottlenecks for

TABLE 6. Sources of errors in gaze estimation systems and literature where they are reported.

	Platforms			
Error Sources	**Head Mounted**	**Desktop (including TV)**	**Automotive**	**Handheld (smartphones, tablets)**
Head pose/ movement	[214][104][21][181] [177][22][9][106] [176][186][184][178] [183][35][109] [117]	[5][65][67][68][152] [6][66][42][44][46] [47][56][48][43][45] [57][54][55][4][148] [167][37][91][81][145] [149][74][150][99][173] [62][63][64][87][71]	[203][162][14][15] [126][159][124][13][121][125] [157][119][16][160][133][131] [128][130]	[108][144][86]
Camera quality/image resolution	[117]	[65][54][49][55] [53][145][75]	[no ref]	[no ref]
Display properties	[194][199]	[54][55]	[129]	[no ref]
User distance/ Viewing angle	[192][154][165][109] [103] [117]	[65][67][68][6][47] [43][45][55][4][169] [37][97][51][63]	----	[164][163][86]
Hand/Platform movement	[114][50]	--	[no ref]	[141][140][86]
Illumination changes	[111]	[5][91][79][87]	[162][133][130]	[no ref]
Occlusion	[105][102]	[5][67][7][169][71]	[126][127][133][128][130]	[no ref]
Human eye conditions/user dependence	[111][115]	[65][152][6][45] [169][145][95][58]	[128][129]	[213]

each platform. We present here the concept of such an evaluation framework in the sections below.

B. CONCEPT OF A PERFORMANCE EVALUATION FRAMEWORK FOR GAZE ESTIMATION SYSTEMS

The framework is to be built around a set of standardized experiments for evaluating various gaze tracking systems. Through the experiments, practical performance limits of a given gaze tracking algorithm or system can be tested under the influence of various parameters: such as variations of head pose, viewing angle, screen size and resolution, eye-occlusion, platform movement and illumination changes. The structure of the framework is outlined in Fig.11. The advantage of having such a framework is that it can answer several critical queries related to eye gaze system design and performance.

For example, which system parameters affect the performance in a particular use-case? How does a particular system perform when compared with similar systems under certain operating conditions or in a particular use-case? Can an algorithm designed for one platform be ported and implemented effectively in another platform? What are the performance bottlenecks of individual algorithms? At present it is challenging to answer such questions as there are no resources that allow us to do practical comparative testing of gaze estimation systems.

C. METHODOLOGY

A typical eye tracking setup comprises of the user, gaze tracker and the tracking environment and each of these components may influence the overall eye tracking performance. A schematic diagram of such a setup and these factors are listed in Fig.11. The proposed experimental framework aims to test impacts of these factors on a tracker's accuracy. A typical ''experiment'' consists of the following steps: a user is asked to sit in front of the eye tracker and their eyes are calibrated for a session. The user is presented with a graphical user interface when the tracker records their gaze coordinates as the user gazes at several points on the screen. The gaze error in degrees is calculated from the shift between ground truth and tracked gaze locations. Some evaluation experiments done and planned with commercial eye trackers are presented here. Preliminary results can be found in [24].

1) ESTIMATING IMPACT OF HEAD POSE

A user is positioned in front of the eye tracker while a video camera captures the position of the user's head simultaneously. The user's head pose in roll, pitch and yaw angles are obtained from the video using an appearance model as shown in Fig.12. Then the user is asked to turn their head to specific fixed positions (in roll pitch yaw angles) and their gaze is tracked on the same interface again. The gaze accuracy scores corresponding to various head pose angles are presented in Fig 13. For a particular tracker it was observed

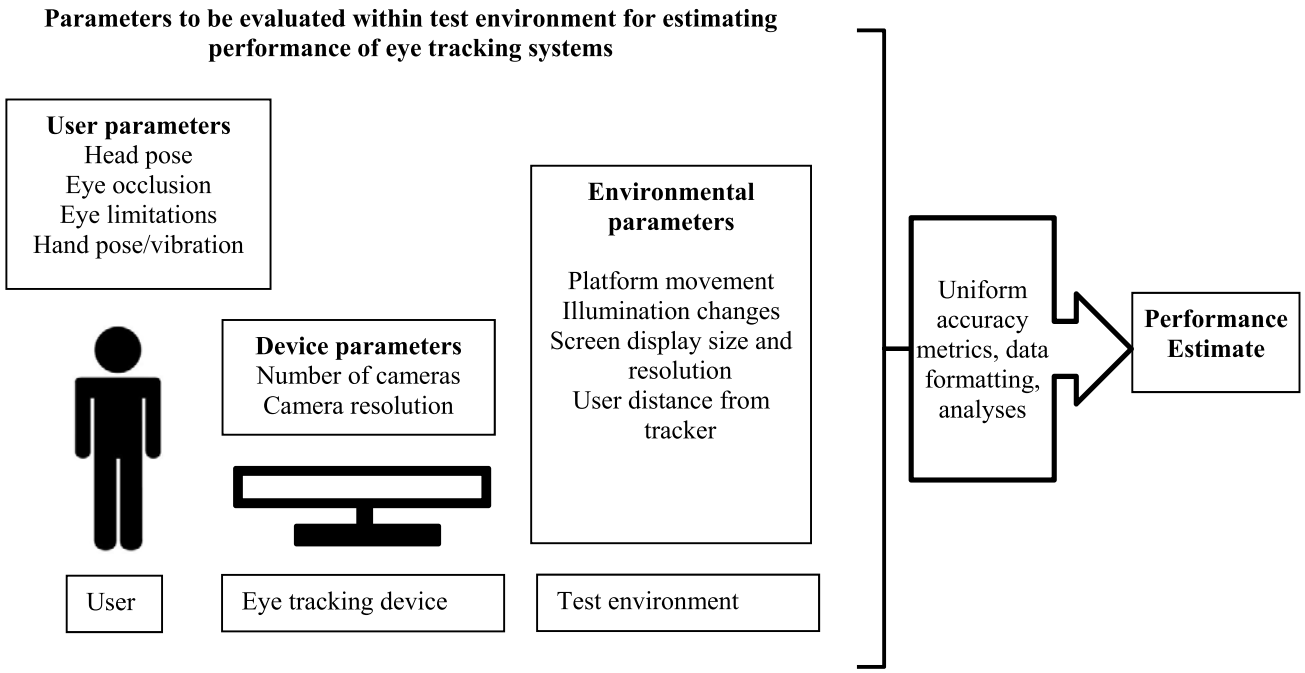


FIGURE 11. Outline concept of a methodological framework for performance evaluation of eye gaze systems.

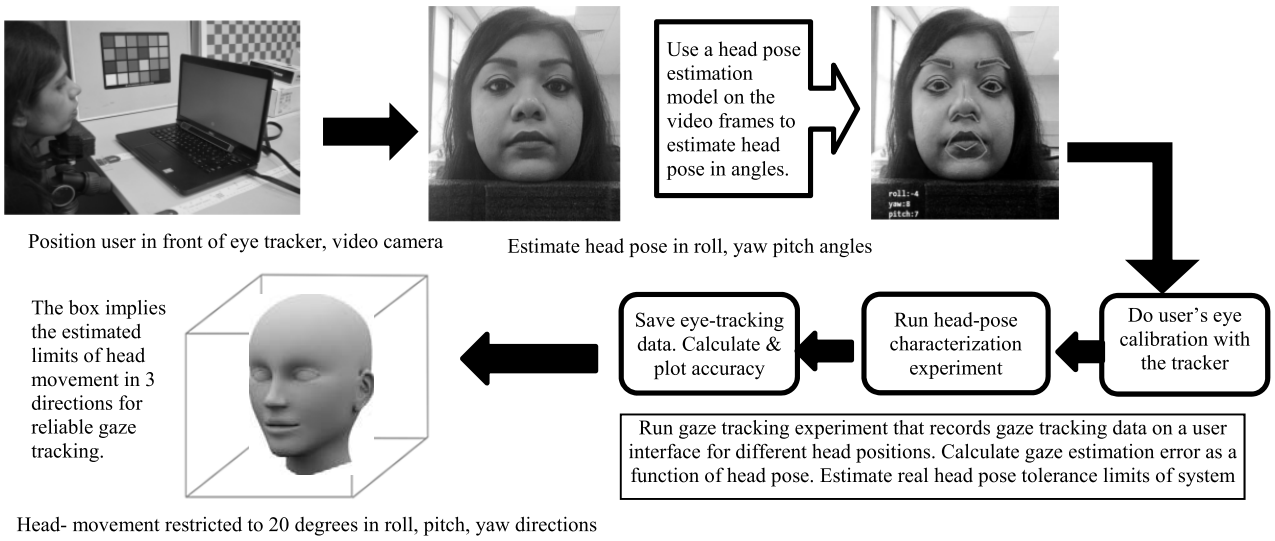


FIGURE 12. Steps of quantifying head pose tolerance limits of a given eye tracking system.

that for reliable gaze tracking, head pose must be restricted within 20 degrees of movement in 3 directions and in this way, the practical head-pose tolerance limits of the tracker could be estimated.

2) USER DISTANCE AND VIEWING ANGLE

For this experiment, the users are positioned at successively increasing distances from the tracker - computer screen setup (40 cm to 100 cm in 15 cm interval) and the gaze tracking accuracy data are recorded for each user position for fixed frontal head poses. The user viewing angle decreases as the user moves away from the screen and the gaze tracking errors are seen to increase with increasing viewing angles. Tracking stops when users are closer than 40 cm.

3) ILLUMINATION LEVELS

Several different illumination levels can be introduced during the experiments. There are cool temperature fluorescents (color temperature ~6400K), warm incandescent lamps (color temperature ~ 2500 K) and mixed lamps (color temperature ~ 5500K) that are available at various intensities (100-3500 lux). Eye tracking experiments will be repeated under these illumination levels to study impact on tracking accuracy.

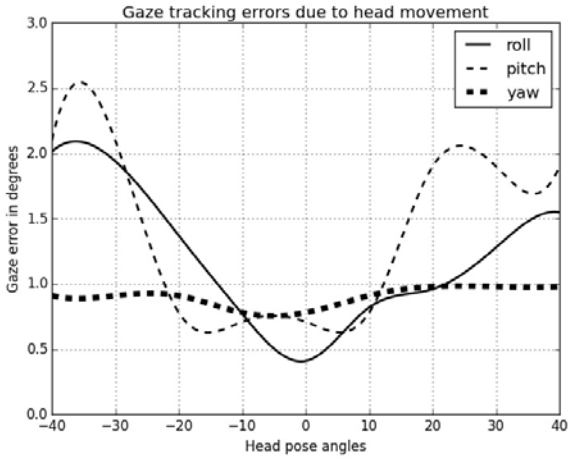


FIGURE 13. The plot shows the gaze tracking errors arising due to user head motion while using a commercial eye tracker during our experiments. The human head has a degree of flexibility in the order of ∼40 degrees of angular movement in each of the roll, pitch and yaw directions. However the plot implies that for reliable gaze tracking with the given tracker, a user needs to keep their head position limited within a fixed angular range in each of these three directions. Through our evaluation process, we estimated this head movement limitation to be equivalent to ∼20 degrees about the central position. If the user's head moves beyond these angles, gaze tracking errors of the tracker rises above acceptable levels. However, another feature observed from this plot is that the impact of head roll and pitch on tracking error are relatively more pronounced than yaw variations.

4) DISPLAY SIZE AND RESOLUTION

The eye tracking experiment is run on displays of various sizes (9, 11, 13.5, 15 inches) and resolutions (800×600, 1024×768, 1280×768, 1366×768) and the corresponding gaze tracking errors are estimated. We have plans for including a 44 inch TV with a specialized tracker intended for gaze tracking for large screens in our testing framework and study impacts of user viewing angles and display sizes.

5) OCCLUSION

We have already observed certain eye trackers having problems with data collection from a user wearing glasses while another tracker has no such issues. Therefore, the level of tolerance of a tracker to user wearing glasses have to be evaluated by operating both trackers simultaneously while doing the gaze experiment for users with and without glasses.

6) PLATFORM MOVEMENTS

The operating conditions faced by a gaze tracker on a static platform like desktop is largely different from that of a dynamic platform like a smartphone or head-mounted setup. We therefore plan to evaluate performance of eye trackers running on such dynamic platforms to realistically observe the difference in performance, study impact of platform movements and other influencing factors.

D. STUDYING DYNAMIC EYE MOVEMENT CHARACTERISTICS

In reality, during a given task the human eyes are constantly moving and therefore sequential eye movements can be studied as a statistical process. Through our framework, we aim to do dynamic eye measurements in a video for studying smooth pursuits besides regular fixations. This is a planned inclusion in which the user will be presented with a 'moving target' for the eyes to follow while capturing video of the eye-movements.

This framework is under development at the moment and preliminary results from some of the experiments have been published by us in [24]. The focus of this paper is mainly to highlight the issue of realistic performance evaluation of eye gaze systems through the literature review. Therefore only limited details about the methodology and implementation of the framework are presented here. A more comprehensive paper on the technical details of the framework is under preparation and will have more information.

VII. CONCLUSION AND FUTURE WORK

Eye gaze estimation is an interdisciplinary area of research and development which has received quite a lot of interest from academic, industrial and general user communities in the last decades owing to the ease of availability of computing and hardware resources and increasing demands for human computer interaction methods. In this paper, a detailed literature review is made on the recent advances in eye gaze research, and information in statistical format is presented to highlight the diversity in various aspects such as platforms, setups, users, algorithms and performance measures existing between different branches of this field.

Several different gaze tracking algorithms and their respective advantages and disadvantages were analyzed in Section III. Currently gaze based HCI systems are capable of achieving high speed input and control operations, leading to their implementation in a variety of user platforms and applications, which were discussed in Section IV. Typically, gaze tracking systems at present are capable of determining 3D point of gaze in real time with unconstrained head movement and achieve around 0.5 degrees of angular resolution. However, limitations arising due to gaze tracking camera quality, random illumination changes, user wearing glasses and platform vibrations are not well characterized in contemporary eye gaze research.

The literature review also raises a major question with respect to the consistency and accuracy that can be obtained from the gaze estimation systems when they operate under real world conditions, if they are not properly evaluated.

A variety of factors may affect eye gaze tracking in different platforms, making their performance unpredictable and ultimately questioning their usability in present and future applications. Effects of head movement, user distance and viewing angle, display properties of the setup are still poorly studied, as discussed in Section V. In their presence, practical system performance may differ significantly from expected values and eye gaze may lose its applicability in different consumer use cases.

Further, there is a clear lack of homogeneity in gaze performance metrics as pointed out in the tables of Section V. Some performance measures used, for example: detection

rate or accuracy percentage is difficult to interpret physically and the variety in reporting formats makes inter-comparisons between different systems and algorithms impossible.

Keeping these in mind, the concept of a performance evaluation framework is proposed that will provide practical performance estimates of gaze tracking systems and adopt a uniform set of accuracy metrics for specifying performance. This is an on-going research activity and the details of the evaluation methods to be included in this framework are currently under development for different gaze estimation platforms, and will be included in a subsequent paper.

ACKNOWLEDGMENT

The authors would like to thank Prof. Christopher Dainty and Dr. Claudia Costache for reviewing and providing their valuable feedback on the manuscript.

REFERENCES

[1] A. O. Mohamed, M. P. da Silva, and V. Courboulay. (2007). *A History of Eye Gaze Tracking, Rapport Interne*. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00215967

[2] A. T. Duchowski, *Behavior Research Methods, Instruments, & Computers*, vol. 34. New York, NY, USA: Springer-Verlag, 2002, p. 455.

[3] C. H. Morimoto and M. R. M. Mimica, "Eye gaze tracking techniques for interactive applications," *Comput. Vis. Image Understand.*, vol. 98, no. 1, pp. 4–24, 2005.

[4] C. H. Morimoto, A. Amir, and M. Flickner, "Detecting eye position and gaze from a single camera and 2 light sources," in *Proc. Int. Conf. Pattern Recogn.*, vol. 4. 2002, pp. 314–317.

[5] P. M. Corcoran, F. Nanu, S. Petrescu, and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," *IEEE Trans. Consum. Electron.*, vol. 58, no. 2, pp. 347–355, May 2012.

[6] D. W. D. Hansen, J. S. J. Agustin, and A. Villanueva, "Homography normalization for robust gaze estimation in uncalibrated setups," in *Proc. Symp. Eye-Tracking Res. Appl.*, vol. 1. Jul. 2015, pp. 13–20.

[7] S. Y. Gwon, C. W. Cho, H. C. Lee, W. O. Lee, and K. R. Park, "Robust eye and pupil detection method for gaze tracking," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 2, p. 98, 2013.

[8] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, "Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, Nov. 2010.

[9] A. Lanata, G. Valenza, A. Greco, and E. P. Scilingo, "Robust head mounted wearable eye tracking system for dynamical calibration," *J. Eye Movement Res.*, vol. 8, no. 5, pp. 1–15, Nov. 2015.

[10] E. S. Kim *et al.*, "Development of an untethered, mobile, low-cost head-mounted eye tracker," in *Proc. Symp. Eye Track. Res. Appl.*, vol. 14. 2014, pp. 247–250.

[11] D. Li, J. Babcock, and D. J. Parkhurst, "openEyes: A low-cost head-mounted eye-tracking solution," in *Proc. Symp. Eye Tracking Res. Appl.*, New York, NY, USA, 2006, pp. 95–100.

[12] W. J. Ryan, A. T. Duchowski, and S. T. Birchfield, "Limbus/pupil switching for wearable eye tracking under variable lighting conditions," in *Proc. ETRA*, New York, NY, USA, 2008, pp. 61–64.

[13] T. Ishikawa, S. Baker, I. Matthews, and T. Kanade, "Passive driver gaze tracking with active appearance models," in *Proc. World Congr. Intell. Transp. Syst.*, 2004, pp. 1–12.

[14] J. P. Batista, "A real-time driver visual attention monitoring system," in *Proc. Conf. IbPRIA*, vol. 3522. Estoril, Portugal, 2005, pp. 200–208.

[15] A. Tawari, K. H. Chen, and M. M. Trivedi, "Where is the driver looking: Analysis of head, eye and iris for robust gaze zone estimation," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 988–994.

[16] Y. Liang, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 340–350, Jun. 2007.

[17] Q. Ji and X. Yang, "Real-time eye, gaze, and face pose tracking for monitoring driver vigilance," *Real-Time Imag.*, vol. 8, no. 5, pp. 357–377, Oct. 2002.

[18] E. Wood and A. Bulling, "EyeTab: Model-based gaze estimation on unmodified tablet computers," in *Proc. ETRA*, New York, NY, USA, 2014, pp. 207–210.

[19] T. Nagamatsu, M. Yamamoto, and H. Sato, "MobiGaze: Development of a gaze interface for handheld mobile devices," in *Proc. Extended Abstracts Hum. Factors Comput. Syst.*, New York, NY, USA, 2010, pp. 3349–3354.

[20] A. Hyrskykari, P. Majaranta, and K.-J. Räihä, "From gaze control to attentive interfaces," in *Proc. HCII*, Las Vegas, NV, USA, Jul. 2005, pp. 1–10.

[21] T. Pfeiffer, "Towards gaze interaction in immersive virtual reality: Evaluation of a monocular eye tracking set-up," in *Proc. Virtuelle Erweiterte Realität-Fünfter Workshop GI-Fachgruppe*, 2008, pp. 81–92.

[22] J. Jimenez, D. Gutierrez, P. Latorre, and U. De Zaragoza, "Gaze-based interaction for virtual environments," *J. Univ. Comput. Sci.*, vol. 14, no. 19, pp. 3085–3098, 2008.

[23] S. Bazrafkan, A. Kar, and C. Costache, "Eye gaze for consumer electronics: Controlling and commanding intelligent systems," *IEEE Consum. Electron. Mag.*, vol. 4, no. 4, pp. 65–71, Oct. 2015.

[24] A. Kar and P. Corcoran, "Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Budapest, Hungary, Oct. 2016, pp. 002061–002066.

[25] A. Kar, S. Bazrafkan, C. Costache, and P. Corcoran, "Eye-gaze systems—-An analysis of error sources and potential accuracy in consumer electronics use cases," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, NV, USA, 2016, pp. 319–320.

[26] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, Mar. 2010.

[27] P. Majaranta and A. Bulling, "Eye tracking and eye-based human-computer interaction," in *Advances in Physiological Computing*. London, U.K.: Springer-Verlag, 2014, pp. 17–39.

[28] A. Poole and L. J. Ball, "Eye tracking in human-computer interaction and usability research: Current status and future prospects," in *Encyclopedia of Human-Computer Interaction*. Pennsylvania, PA, USA: Idea Group, Inc, 2005, pp. 211–219.

[29] J. H. Goldberg and X. P. Kotval, "Computer interface evaluation using eye movements: Methods and constructs," *Int. J. Ind. Ergon.*, vol. 24, no. 6, pp. 631–645, Oct. 1999.

[30] D. D. Salvucci and J. H. Goldberg, "Identifying fixations and saccades in eye-tracking protocols," in *Proc. Eye Tracking Res. Appl. Symp.*, 2000, pp. 71–78.

[31] B. B. Velichkovsky, M. A. Rumyantsev, and M. A. Morozov, "New solution to the midas touch problem: Identification of visual commands via extraction of focal fixations," *Procedia Comput. Sci.*, vol. 39, pp. 75–82, Dec. 2014.

[32] E. D. Guestrin and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1124–1133, Jun. 2006.

[33] H. C. Lee *et al.*, "Remote gaze tracking system on a large display," *Sensors*, vol. 13, no. 10, pp. 13439–13463, 2013.

[34] O. Špakov and D. Miniotas, "Gaze-based selection of standard-size menu items," in *Proc. 7th Int. Conf. Multimodal Interfaces*, New York, NY, USA, 2005, pp. 124–128

[35] L. E. Sibert and R. J. K. Jacob, "Evaluation of eye gaze interaction," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, New York, NY, USA, 2000, pp. 281–288.

[36] M. Kumar, A. Paepcke, and T. Winograd, "EyePoint," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, vol. 7. 2007, p. 421.

[37] Z. Zhu and Q. Ji, "Novel eye gaze tracking techniques under natural head movement," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 12, pp. 2246–2260, Dec. 2007.

[38] K. Harezlak, P. Kasprowski, and M. Stasch, "Towards accurate eye tracker calibration—Methods and procedures," *Procedia Comput. Sci.*, vol. 35, pp. 1073–1081, Sep. 2014.

[39] S. O. Ba and J.-M. Odobez, "Multiperson visual focus of attention from head pose and meeting contextual cues," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 101–116, Jan. 2011.

[40] *Accuracy and Precision Test Method for Remote Eye Trackers, Test Specification Version: 2.1.1*, Tobii Technology AB, Sweden, Feb. 2011.

[41] P. Blignaut, "Mapping the pupil-glint vector to gaze coordinates in a simple video-based eye tracker," *J. Eye Movement Res.*, vol. 7, no. 1, pp. 1–11, 2013.

[42] Z. R. Cherif, A. Nait-Ali, J. F. Motsch, and M. O. Krebs, "An adaptive calibration of an infrared light device used for gaze tracking," in *Proc. 19th IEEE Instrum. Meas. Technol. Conf.*, vol. 2. May 2002, pp. 1029–1033.

[43] X. L. C. Brolly and J. B. Mulligan, "Implicit calibration of a remote gaze tracker," in *Proc. Conf. Comput. Vis. Pattern Recogn. Workshop*, Jun. 2004, p. 134.

[44] R. J. J. Cerrolaza, A. Villanueva, and R. Cabeza, "Taxonomic study of polynomial regressions applied to the calibration of video-oculographic systems," in *Proc. Symp. Eye Tracking Res. Appl.*, vol. 1. 2008, pp. 259–266.

[45] Z. Zhu and Q. Ji, "Eye and gaze tracking for interactive graphic display," *Mach. Vis. Appl.*, vol. 15, no. 3, pp. 139–148, Jul. 2004.

[46] C. Jian-Nan, Z. Chuang, Y. Yan-Tao, L. Yang, and Z. Han, "Eye gaze calculation based on nonlinear polynomial and generalized regression neural network," in *Proc. 5th Int. Conf. Natural Comput.*, Aug. 2009, pp. 617–623.

[47] C. Ma, K.-A. Choi, B.-D. Choi, and S.-J. Ko, "Robust remote gaze estimation method based on multiple geometric transforms," *Opt. Eng.*, vol. 54, no. 8, p. 83103, Aug. 2015.

[48] Z. Zhu, Q. Ji, and K. P. Bennett, "Nonlinear eye gaze mapping function estimation via support vector regression," in *Proc. 18th Int. Conf. Pattern Recogn.*, Hong Kong, China, Aug. 2006, pp. 1132–1135.

[49] J. Zhu and J. Yang, "Subpixel eye gaze tracking," in *Proc. 5th IEEE Int. Conf. Autom. Face Gesture Recogn.*, May 2002, pp. 131–136.

[50] J. Wang, G. Zhang, and J. Shi, "2D gaze estimation based on pupil-glint vector using an artificial neural network," *Appl. Sci.*, vol. 6, no. 6, p. 174, 2016.

[51] Y.-G. Shin, K.-A. Choi, S.-T. Kim, C.-H. Yoo, and S.-J. Ko, "A novel 2-D mapping-based remote eye gaze tracking method using two IR light sources," in *Proc. IEEE Int. Conf. Consum. Electron.*, Las Vegas, NV, USA, Jan. 2015, pp. 190–191.

[52] J. Park, T. Jung, and K. Yim, "Implementation of an eye gaze tracking system for the disabled people," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl.*, Gwangiu, South Korea, Mar. 2015, pp. 904–908.

[53] A. Meyer, M. Böhme, T. Martinetz, and E. Barth, "A single-camera remote eye tracker," *Perception and Interactive Technologies* (Lecture Notes in Computer Science), vol. 4021. New York, NY, USA: Springer, 2006, pp. 208–211.

[54] C. Hennessey, B. Noureddin, and P. Lawrence, "A single camera eye-gaze tracking system with free head motion," *Measurement*, vol. 1, pp. 27–29, Mar. 2006.

[55] C.-C. Lai, S.-W. Shih, and Y.-P. Hung, "Hybrid method for 3-D gaze tracking using glint and contour features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 1, pp. 24–37, Jan. 2015.

[56] T. Ohno and N. Mukawa, "A free-head, simple calibration, gaze tracking system that enables gaze-based interaction," in *Proc. Eye Tracking Res. Appl. Symp. Eye Tracking Res. Appl.*, 2004, pp. 115–122.

[57] D. Beymer and M. Flickner, "Eye gaze tracking using an active stereo head," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, vol. 2. Jun. 2003, pp. II-451–II-458.

[58] S.-W. Shih and J. Liu, "A novel approach to 3-D gaze tracking using stereo cameras.," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 234–245, Feb. 2004.

[59] T. Nagamatsu, Y. Iwamoto, J. Kamahara, N. Tanaka, and M. Yamamoto, "Gaze estimation method based on an aspherical model of the cornea: Surface of revolution about the optical axis of the eye," in *Proc. Symp. Eye-Tracking Res. Appl.*, New York, NY, USA, 2010, pp. 255–258.

[60] T. Nagamatsu, J. Kamahara, and N. Tanaka, "Calibration-free gaze tracking using a binocular 3D eye model," in *Proc. Extended Abstracts Hum. Factors Comput. Syst.*, New York, NY, USA, 2009, pp. 3613–3618.

[61] D. Model and M. Eizenman, "User-calibration-free remote eye-gaze tracking system with extended tracking range," in *Proc. 24th Can. Conf. Elect. Comput. Eng.*, Niagara Falls, ON, Canada, 2011, pp. 001268–001271.

[62] K. Wang and Q. Ji, "Real time eye gaze tracking with Kinect," in *Proc. 23rd Int. Conf. Pattern Recogn.*, Cancún, Mexico, 2016, pp. 2752–2757.

[63] X. Zhou, H. Cai, Z. Shao, H. Yu, and H. Liu, "3D eye model-based gaze estimation from a depth sensor," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Qingdao, China, Dec. 2016, pp. 369–374.

[64] L. Jianfeng and L. Shigang, "Eye-model-based gaze estimation by RGB-D camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, Columbus, OH, USA, Jun. 2014, pp. 606–610.

[65] J. Huang, Q. Cai, Z. Liu, N. Ahuja, and Z. Zhang, "Towards accurate and robust cross-ratio based gaze trackers through learning from simulation," in *Proc. ETRA*, 2014, pp. 75–82.

[66] D. H. Yoo and M. J. Chung, "A novel non-intrusive eye gaze estimation using cross-ratio under large head motion," *Comput. Vis. Image Understand.*, vol. 98, no. 1, pp. 25–51, Apr. 2005.

[67] F. L. Coutinho and C. H. Morimoto, "Free head motion eye gaze tracking using a single camera and multiple light sources," in *Proc. 19th Brazilian Symp. Comput. Graph. Image Process.*, 2006, pp. 171–178.

[68] F. L. Coutinho and C. H. Morimoto, "Augmenting the robustness of cross-ratio gaze tracking methods to head movement," in *Proc. Symp. Eye Tracking Res. Appl.*, 2012, pp. 59–66.

[69] J. J. Kang, E. D. Guestrin, W. J. Maclean, and M. Eizenman, "Simplifying the cross-ratios method of point-of-gaze estimation," in *Proc. 30th Can. Med. Biol. Eng. Conf.*, 2007, pp. 1–4.

[70] I. Bacivarov, M. Ionita, and P. Corcoran, "Statistical models of appearance for eye tracking and eye-blink detection and measurement," *IEEE Trans. Consum. Electron.*, vol. 54, no. 3, pp. 1312–1328, Aug. 2008.

[71] P. Koutras and P. Maragos, "Estimation of eye gaze direction angles based on active appearance models," in *Proc. IEEE Int. Conf. Image Process.*, Quebec City, QC, Canada, Sep. 2015, pp. 2424–2428.

[72] F. Lu, Y. Gao, and X. Chen, "Estimating 3D gaze directions using unlabeled eye images via synthetic iris appearance fitting," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1772–1782, Sep. 2016.

[73] Y.-L. Wu, C.-T. Yeh, W.-C. Hung, and C.-Y. Tang, "Gaze direction estimation using support vector machine with active appearance model," *Multimed. Tools Appl.*, pp. 1–26, 2012.

[74] H. L. H. Lu, C. W. C. Wang, and Y. C. Y. Chen, "Gaze tracking by binocular vision and LBP features," in *Proc. 19th Int. Conf. Pattern Recogn.*, Tampa, FL, USA, 2008, pp. 1–4.

[75] C. M. Yilmaz and C. Kose, "Local binary pattern histogram features for on-screen eye-gaze direction estimation and a comparison of appearance based methods," in *Proc. 39th Int. Conf. Telecommun. Signal Process.*, Vienna, Austria, 2016, pp. 693–696.

[76] S. Chen and C. Liu, "Eye detection using discriminatory Haar features and a new efficient SVM," *Image Vis. Comput.*, vol. 33, pp. 68–77, Jan. 2015.

[77] Y. Li, X. Xu, N. Mu, and L. Chen, "Eye-gaze tracking system by haar cascade classifier," in *Proc. IEEE 11th Conf. Ind. Electron. Appl.*, Hefei, China, Jun. 2016, pp. 564–567.

[78] S. Baluja and D. Pomerleau, "Non-intrusive gaze tracking using artificial neural networks," Carnegie Mellon Univ. Pittsburgh, Pittsburgh, PA, USA, Tech. Rep. FS-93-04, 1994, pp. 753–760.

[79] T. Schneider, B. Schauerte, and R. Stiefelhagen, "Manifold alignment for person independent appearance-based gaze estimation," in *Proc. 22nd Int. Conf. Pattern Recogn.*, 2014, pp. 1167–1172.

[80] L. Yu, J. Xu, and S. Huang, "Eye-gaze tracking system based on particle swarm optimization and BP neural network," in *Proc. 12th World Congr. Intell. Control Autom.*, Guilin, China, 2016, pp. 1269–1273.

[81] C. C. Lai, Y. T. Chen, K. W. Chen, S. C. Chen, S. W. Shih, and Y. P. Hung, "Appearance-based gaze tracking with free head movement," in *Proc. Int. Conf. Pattern Recogn.*, vol. 1. 2014, pp. 1869–1873.

[82] D. Benavides and D. L. Borges, "Eye detection in unrestrained settings using efficient match kernels and SVM classification," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Budapest, Hungary, 2016, pp. 002801–002806.

[83] H. Huang and J. Wechsler, "Eye location using genetic algorithm," in *Proc. 2nd Int. Conf. Audio Video-Based Biometric Pers. Authentication*, 1999, pp. 130–135.

[84] A. George and A. Routray, "Real-time eye gaze direction classification using convolutional neural network," in *Proc. Int. Conf. Signal Process. Commun.*, Bengaluru, India, 2016, pp. 1–5.

[85] R. Konrad, *Near-Eye Display Gaze Tracking Via Convolutional Neural Networks*. [Online]. Available: https://web.stanford.edu/class/cs231a/prev_projects_2016/eye-display-gaze-2.pdf

[86] K. Krafka *et al.*, "Eye tracking for everyone," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Las Vegas, NV, USA, Jun. 2016, pp. 2176–2184.

[87] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, Jun. 2015, pp. 4511–4520.

[88] W. Wang, Y. Huang, and R. Zhang, "Driver gaze tracker using deformable template matching," in *Proc. IEEE Int. Conf. Veh. Electron. Safety*, Jul. 2011, pp. 244–247.

[89] M. J. T. Reinders, "Eye tracking by template matching using an automatic codebook generation scheme," in *Proc. 3rd Annu. Conf. Adv. School Comput. Imag.*, Heijen, The Netherlands, Jun. 1997, pp. 215–221.

[90] S. Ramadan, W. Abd-Almageed, and C. Smith, "Eye tracking using active deformable models," in *Proc. 3rd Indian Conf. Comput. Vis., Graph. Image Process.*, India, 2002.

[91] I. F. Ince and J. W. Kim, "A 2D eye gaze estimation system with low-resolution webcam images," *EURASIP J. Adv. Signal Process.*, vol. 2011, no. 1, p. 40, Dec. 2011.

[92] S. Zhai, C. Morimoto, and S. Ihde, "Manual and gaze input cascaded (MAGIC) pointing," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, New York, NY, USA, 1999, pp. 246–253.

[93] M. U. Ghani, S. Chaudhry, M. Sohail, and M. N. Geelani, "GazePointer: A real time mouse pointer control implementation based on eye gaze tracking," in *Proc. 16th Int. Multi Topic Conf.*, 2013, pp. 154–159.

[94] J. S. Agustin, J. C. Mateo, J. P. Hansen, and A. Villanueva, "Evaluation of the potential of gaze input for game interaction," *PsychNology J.*, vol. 7, no. 2, pp. 213–236, 2009.

[95] P. Kasprowski and K. Harężlak, "Cheap and Easy PIN Entering Using Eye Gaze," *Ann. UMCS, Inf.*, vol. 14, no. 1, pp. 75–83, 2014.

[96] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd, "Reducing shoulder-surfing by using gaze-based password entry," in *Proc. 3rd Symp. Usable Privacy Secur.*, vol. 7. 2007, pp. 13–19.

[97] A. Bulling, F. Alt, and A. Schmidt, "Increasing the security of gaze-based cued-recall graphical passwords using saliency masks," in *Proc. ACM Annu. Conf. Hum. Factors Comput. Syst.*, vol. 12. 2012, p. 3011.

[98] S. T. Iqbal and B. P. Bailey, "Using eye gaze patterns to identify user tasks," in *Grace Hopper Celebration of Women in Computing*. Lausanne, Switzerland: Frontiers, 2004, p. 6.

[99] A. Doshi and M. M. Trivedi, "Head and gaze dynamics in visual attention and context learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, Jun. 2009, pp. 77–84.

[100] J. W. Lee, C. W. Cho, K. Y. Shin, E. C. Lee, and K. R. Park, "3D gaze tracking method using Purkinje images on eye optical model and pupil," *Opt. Lasers Eng.*, vol. 50, no. 5, pp. 736–751, 2012.

[101] W. W. Abbott and A. A. Faisal, "Ultra-low-cost 3D gaze estimation: An intuitive high information throughput compliment to direct brain-machine interfaces," *J. Neural Eng.*, vol. 9, no. 4, p. 46016, 2012.

[102] E. Schneider *et al.*, "Gaze-aligned head-mounted camera with pan, tilt, and roll motion control for medical documentation and teaching applications," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Taipei, Taiwan, Oct. 2006, pp. 327–331.

[103] M. Y. Kim, S. Yang, and D. Kim, "Head-mounted binocular gaze detection for selective visual recognition systems," *Sens. Actuators A, Phys.*, vol. 187, pp. 29–36, Nov. 2012.

[104] K. Takemura, K. Takahashi, J. Takamatsu, and T. Ogasawara, "Estimating 3-D point-of-regard in a real environment using a head-mounted eye-tracking system," *IEEE Trans. Human-Mach. Syst.*, vol. 44, no. 4, pp. 531–536, Aug. 2014.

[105] X. Long, O. K. Tonguz, and A. Kiderman, "A high speed eye tracking system with robust pupil center estimation algorithm," in *Proc. 29th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Lyon, France, Aug. 2007, pp. 3331–3334.

[106] E. Schneider, T. Dera, K. Bard, S. Bardins, G. Boening, and T. Brand, "Eye movement driven head-mounted camera: It looks where the eyes look," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, vol. 3. Oct. 2005, pp. 2437–2442.

[107] S. H. Lee, J.-Y. Lee, and J. S. Choi, "Design and implementation of an interactive HMD for wearable AR system," in *Proc. 17th Korea-Jpn. Joint Workshop Frontiers Comput. Vis.*, Ulsan, South Korea, 2011, pp. 1–6.

[108] E. C. Lee, Y. J. Ko, and K. R. Park, "Gaze tracking based on active appearance model and multiple support vector regression on mobile devices," *Opt. Eng.*, vol. 48, no. 7, pp. 077002-1–077002-11, Jul. 2009.

[109] T. Toyama, A. Dengel, W. Suzuki, and K. Kise, "Wearable reading assist system: Augmented reality document combining document retrieval and eye tracking," in *Proc. Int. Conf. Doc. Anal. Recogn.*, 2013, pp. 30–34.

[110] T. Piumsomboon, G. Lee, R. W. Lindeman, and M. Billinghurst, "Exploring natural eye-gaze-based interaction for immersive virtual reality," in *Proc. IEEE Symp. 3D User Interfaces*, Los Angeles, CA, USA, Mar. 2017, pp. 36–39.

[111] M. Niener, M. Stamminger, J. Thies, C. Theobalt, and M. Zollhöfer. (2016). "FaceVR: Real-time facial reenactment and eye gaze control in virtual reality." [Online]. Available: https://arxiv.org/abs/1610.03151

[112] N. Sidorakis, G. A. Koulieris, and K. Mania, "Binocular eye-tracking for the control of a 3D immersive multimedia user interface," in *Proc. IEEE 1st Workshop Everyday Virtual Reality*, Arles, France, Mar. 2015, pp. 15–18.

[113] A. M. Soccini, "Gaze estimation based on head movements in virtual reality applications using deep learning," in *Proc. IEEE Virtual Reality*, Los Angeles, CA, USA, Mar. 2017, pp. 413–414.

[114] A. Plopski, J. Orlosky, Y. Itoh, C. Nitschke, K. Kiyokawa, and G. Klinker, "Automated spatial calibration of HMD systems with unconstrained eye-cameras," in *Proc. IEEE Int. Symp. Mixed Augmented Reality*, Mérida, Mexico, Sep. 2016, pp. 94–99.

[115] J. Y. Lee, H. M. Park, S. H. Lee, T. E. Kim, and J. S. Choi, "Design and implementation of an augmented reality system using gaze interaction," in *Proc. Int. Conf. Inf. Sci. Appl.*, Jeju Island, 2011, pp. 1–8.

[116] M. Bâce, T. Leppänen, D. G. de Gomez, and A. R. Gomez, "ubiGaze: Ubiquitous augmented reality messaging using gaze gestures," in *Proc. SIGGRAPH ASIA Mobile Graph. Interact. Appl.*, 2016, pp. 11:1–11:5.

[117] J. Orlosky, T. Toyama, K. Kiyokawa, and D. Sonntag, "ModulAR: Eye-controlled vision augmentations for head mounted displays," *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 11, pp. 1259–1268, Nov. 2015.

[118] X. Liu, F. Xu, and K. Fujimura, "Real-time eye detection and tracking for driver observation under various light conditions," in *Proc. IEEE Intell. Vehicle Symp.*, vol. 2. Jun. 2002, pp. 344–351.

[119] S. J. Lee, J. Jo, H. G. Jung, K. R. Park, and J. Kim, "Real-time gaze estimator based on driver's head orientation for forward collision warning system," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 254–267, Mar. 2011.

[120] M.-C. Chuang, R. Bala, E. A. Bernal, P. Paul, and A. Burry, "Estimating gaze direction of vehicle drivers using a smartphone camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, Jun. 2014, pp. 165–170.

[121] A. Tawari and M. M. Trivedi, "Robust and continuous estimation of driver gaze zone by dynamic analysis of multiple face videos," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 344–349.

[122] J. H. Oh and N. Kwak, "Recognition of a Driver's gaze for vehicle headlamp control," *IEEE Trans. Veh. Technol.*, vol. 61, no. 5, pp. 2008–2017, Jun. 2012.

[123] D. D. Salvucci and A. Liu, "The time course of a lane change: Driver control and eye-movement behavior," *Transp. Res. F, Traffic Psychol. Behaviour*, vol. 5, no. 2, pp. 123–132, 2002.

[124] I. H. Choi and Y. G. Kim, "Head pose and gaze direction tracking for detecting a drowsy driver," *Appl. Math. Inf. Sci.*, vol. 9, no. 2, pp. 505–512, 2015.

[125] Q. Ji, Z. Zhu, and P. Lan, "Real-time nonintrusive monitoring and prediction of driver fatigue," *IEEE Trans. Veh. Technol.*, vol. 53, no. 4, pp. 1052–1068, Jul. 2004.

[126] L. M. Bergasa and J. Nuevo, "Real-time system for monitoring driver vigilance," in *Proc. IEEE Int. Symp. Ind. Electron.*, vol. 3. Jun. 2005, pp. 1303–1308.

[127] X. Sun, L. Xu, and J. Yang, "Driver fatigue alarm based on eye detection and gaze estimation," *Proc. SPIE*, vol. 6786, pp. 678612-1–678612-6, Nov. 2007.

[128] F. Vicente, Z. Huang, X. Xiong, F. D. L. Torre, W. Zhang, and D. Levi, "Driver gaze tracking and eyes off the road detection system," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2014–2027, Aug. 2015.

[129] R. Zheng, K. Nakano, H. Ishiko, K. Hagita, M. Kihira, and T. Yokozeki, "Eye-gaze tracking analysis of driver behavior while interacting with navigation systems in an urban area," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 4, pp. 546–556, Aug. 2016.

[130] O. Stan, L. Miclea, and A. Centea, "Eye-gaze tracking method driven by raspberry PI applicable in automotive traffic safety," in *Proc. 2nd Int. Conf. Artif. Intell., Modelling Simulation*, Madrid, Spain, 2014, pp. 126–130.

[131] T. Kowsari, S. S. Beauchemin, M. A. Bauer, D. Laurendeau, and N. Teasdale, "Multi-depth cross-calibration of remote eye gaze trackers and stereoscopic scene systems," in *Proc. IEEE Intell. Vehicles Symp.*, Dearborn, MI, USA, Jun. 2014, pp. 1245–1250.

[132] S. Jha and C. Busso, *Analyzing the Relationship Between Head Pose and Gaze to Model Driver Visual Attention*. Rio de Janeiro, Brazil: IEEE, 2016, pp. 1–6.
[133] L. Fridman, J. Lee, B. Reimer, and T. Victor, "'Owl' and 'Lizard': Patterns of head pose and eye pose in driver gaze classification," *IET Comput. Vis.*, vol. 10, no. 4, pp. 308–313, 2016.
[134] V. Vaitukaitis and A. Bulling, "Eye gesture recognition on portable devices," in *Proc. ACM Conf. Ubiquitous Comput.*, vol. 12. 2012, p. 711.
[135] K. Lukander, "A system for tracking gaze on handheld devices," *Behavior Res. Meth.*, vol. 38, no. 4, pp. 660–666, 2006.
[136] C. Holland, A. Garza, E. Kurtova, J. Cruz, and O. Komogortsev, "Usability evaluation of eye tracking on an unmodified common tablet," in *Proc. Extended Abstracts Hum. Factors Comput. Syst.*, vol. 13. 2013, p. 295.
[137] T. Imabuchi, O. Dicky, A. Prima, H. Kikuchi, Y. Horie, and H. Ito, "Visible-spectrum Remote Eye Tracker for Gaze Communication," *Proc. SPIE*, vol. 9443, p. 944333, Mar. 2015.
[138] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. Comput. Vis. Pattern Recogn.*, vol. 1. 2001, pp. I-511–I-518.
[139] H. Elleuch, A. Wali, and A. M. Alimi, "Smart tablet monitoring by a real-time head movement and eye gestures recognition system," in *Proc. Int. Conf. Future Internet Things Cloud*, Barcelona, Spain, 2014, pp. 393–398.
[140] C. Pino and I. Kavasidis, "Improving mobile device interaction by eye tracking analysis," in *Proc. Comput. Sci. Inf.*, 2012, pp. 1199–1202.
[141] H. Drewes and A. Schmidt, "Interacting with the computer using gaze gestures," in *Proc. Int. Conf. Hum.-Comput. Interact.*, 2007, pp. 475–488.
[142] E. Miluzzo, T. Wang, and A. T. Campbell, "EyePhone: Activating mobile phones with your eyes," in *Proc. Workshop Netw., Syst., Appl. Mobile Handhelds*, 2010, pp. 15–20.
[143] D. Liu, B. Dong, X. Gao, and H. Wang, "Exploiting eye tracking for smartphone authentication," in *Applied Cryptography and Network Security* (Lectures Notes Computer Science), vol. 9092, T. Malkin, V. Kolesnikov, A. Lewko, and M. Polychronakis, Eds. New York, NY, USA: Springer, 2015.
[144] Y.-W. Sun, C.-K. Chiang, and S.-H. Lai, "Integrating eye tracking and motion sensor on mobile phone for interactive 3D display," *Proc. SPIE*, vol. 8856, p. 88560F, Sep. 2013.
[145] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, "Adaptive linear regression for appearance-based gaze estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 2033–2046, Oct. 2014.
[146] A. Villanueva, R. Cabeza, and S. Porta, "Eye tracking: Pupil orientation geometrical modeling," *Image Vis. Comput.*, vol. 24, no. 7, pp. 663–679, 2006.
[147] S.-W. Shih, Y.-T. Wu, and J. Liu, "A calibration-free gaze tracking technique," in *Proc. Int. Conf. Pattern Recogn.*, vol. 4. Sep. 2000, pp. 201–204.
[148] R. Newman, Y. Matsumoto, S. Rougeaux, and A. Zelinsky, "Real-time stereo tracking for head pose and gaze estimation," in *Proc. 4th IEEE Int. Conf. Autom. Face Gesture Recogn.*, Grenoble, France, Mar. 2000, pp. 122–128.
[149] Y. Sugano, Y. Matsushita, and Y. Sato, "Learning-by-synthesis for appearance-based 3D gaze estimation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, Jun. 2014, pp. 1821–1828.
[150] K. Tan, D. J. Kriegman, and N. Ahuja, "Appearance-based eye gaze estimation," in *Proc. WACV*, 2002, pp. 191–195.
[151] X. Zhao, X. Zou, and Z. Chi, "A 3D gaze estimation method based on facial feature tracking," in *Proc. Int. Conf. Comput. Healthcare*, 2012, pp. 9–12.
[152] Z. Zhang and Q. Cai, "Improving cross-ratio-based eye tracking techniques by leveraging the binocular fixation constraint," in *Proc. Symp. Eye Tracking Res. Appl.*, vol. 14. 2014, pp. 267–270.
[153] C. Yu, M. Scheutz, and P. Schermerhorn, "Investigating multimodal real-time patterns of joint attention in an HRI word learning task," in *Proc. 5th ACM/IEEE Int. Conf. Hum.-Robot Interact.*, 2010, pp. 309–316.
[154] V. Rantanen *et al.*, "A wearable, wireless gaze tracker with integrated selection command source for human-computer interaction," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, no. 5, pp. 795–801, Sep. 2011.
[155] M. Stengel, S. Grogorick, M. Eisemann, E. Eisemann, and M. A. Magnor, "An affordable solution for binocular eye tracking and calibration in head-mounted displays," in *Proc. 23rd ACM Int. Conf. Multimedia*, New York, NY, USA, 2015, pp. 15–24.
[156] B. R. Pires, M. Hwangbo, M. Devyver, and T. Kanade, "Visible-spectrum gaze tracking for sports," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. Workshops*, Jun. 2013, pp. 1005–1010.
[157] L. Fletcher and A. Zelinsky, "Driver inattention detection based on eye gaze—Road event correlation," *Int. J. Robot. Res.*, vol. 28, no. 6, pp. 774–801, 2009.
[158] K. Yamashiro *et al.*, "Automatic calibration of an in-vehicle gaze tracking system using driver's typical gaze behavior," in *Proc. IEEE Intell. Vehicles Symp.*, Xi'an, China, Jun. 2009, pp. 998–1003.
[159] A. Doshi and M. M. Trivedi, "On the roles of eye gaze and head dynamics in predicting driver's intent to change lanes," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 453–462, Sep. 2009.
[160] P. Smith, M. Shah, and N. da Vitoria Lobo, "Determining driver visual attention with one camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 4, pp. 205–218, Dec. 2003.
[161] J. Wu and M. M. Trivedi, "Simultaneous eye tracking and blink detection with interactive particle filters," *EURASIP J. Adv. Signal Process.*, vol. 2008, Jan. 2008, Art. no. 114.
[162] R. Oyini Mbouna, S. G. Kong, and M.-G. Chun, "Visual analysis of eye state and head pose for driver alertness monitoring," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1462–1469, Sep. 2013.
[163] I. Lissoboi and H. Kasai, "Development of an efficient method for eye detection on mobile CE devices," in *Proc. Int. Symp. Comput. Consum. Control*, 2012, pp. 337–340.
[164] K. Shibasato, C. Tateyama, H. Ohtsuka, and Y. Shimada, "Implementation of application by gaze interaction on a tablet computer for challenged people," in *Proc. 9th Int. Conf. Sens. Technol.*, Auckland, New Zealand, 2015, pp. 369–373.
[165] L. H. Yu and M. Eizenman, "A new methodology for determining point-of-gaze in head-mounted eye tracking systems," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 10, pp. 1765–1773, Oct. 2004.
[166] C. Morimoto, D. Koons, A. Amir, and M. Flickner, "Pupil detection and tracking using multiple. Light sources," *Image Vis. Comput.*, vol. 18, no. 4, pp. 331–335, 2000.
[167] S. J. Baek, K. A. Choi, C. Ma, Y. H. Kim, and S. J. Ko, "Eyeball model-based iris center localization for visible image-based eye-gaze tracking systems," *IEEE Trans. Consum. Electron.*, vol. 59, no. 2, pp. 415–421, May 2013.
[168] F. Lu, T. Okabe, Y. Sugano, and Y. Sato, "A head pose-free approach for appearance-based gaze estimation," in *Proc. Brit. Mach. Vis. Conf.*, 2011, pp. 126.1–126.11.
[169] A. Nakazawa and C. Nitschke, "Point of gaze estimation through corneal surface reflection in an active illumination environment," in *Computer Vision* (Lecture Notes in Computer Science), vol. 7573. New York, NY, USA: Springer-Verlag, 2012, pp. 159–172.
[170] D. LeBlanc, A. Forget, and R. Biddle, "Guessing click-based graphical passwords by eye tracking," in *Proc. 8th Int. Conf. Privacy, Secur. Trust*, 2010, pp. 197–204.
[171] N. M. M. Hassan and W. Mansor, "Detection of eye movements for controlling a television," in *Proc. IEEE 10th Int. Colloq. Signal Process. Appl.*, Mar. 2014, pp. 257–260.
[172] R. Alonso, M. Causse, F. Vachon, R. Parise, F. Dehais, and P. Terrier, "Evaluation of head-free eye tracking as an input device for air traffic control," *Ergonomics*, vol. 56, no. 2, pp. 246–255, 2013.
[173] J. Weaver, K. Mock, and B. Hoanca, "Gaze-based password authentication through automatic clustering of gaze points," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2011, pp. 2749–2754.
[174] K. R. Newman and C. R. Sears, "Eye gaze tracking reveals different effects of a sad mood induction on the attention of previously depressed and never depressed women," *Cognit. Therapy Res.*, vol. 39, no. 3, pp. 292–306, 2015.
[175] B. Noris, J. B. Keller, and A. Billard, "A wearable gaze tracking system for children in unconstrained environments," *Comput. Vis. Image Understand.*, vol. 115, no. 4, pp. 476–486, 2011.
[176] A. Carbone, F. Martínez, E. Pissaloux, D. Mazeika, and R. Velázquez, "On the design of a low cost gaze tracker for interaction," *Procedia Technol.*, vol. 3, pp. 89–96, 2012.
[177] Z. Ye *et al.*, "Detecting eye contact using wearable eye-tracking glasses," in *Proc. ACM Conf. Ubiquitous Comput.*, vol. 12. 2012, p. 699.
[178] N. Kumar, S. Kohlbecher, and E. Schneider, "A novel approach to video-based pupil tracking," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2009, pp. 1255–1262.

[179] S. H. Kwon and M. Y. Kim, "Selective attentional point-tracking through a head-mounted stereo gaze tracker based on trinocular epipolar geometry," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, May 2015, pp. 1617–1621.
[180] X. Li and W. G. Wee, "An efficient method for eye tracking and eye-gazed fov estimation," in *Proc. Int. Conf. Image Process.*, 2009, pp. 2597–2600.
[181] A. Lanata, A. Greco, G. Valenza, and E. P. Scilingo, "On the tridimensional estimation of the gaze point by a stereoscopic wearable eye tracker," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Aug. 2015, pp. 2283–2286.
[182] D. J. Mack, P. Schönle, S. Fateh, T. Burger, Q. Huang, and U. Schwarz, "An EOG-based, head-mounted eye tracker with 1 kHz sampling rate," in *Proc. IEEE Biomed. Circuits Syst. Conf.*, Atlanta, GA, USA, Oct. 2015, pp. 1–4.
[183] H. Heo, E. Lee, K. Park, C. Kim, and M. Whang, "A realistic game system using multi-modal user interfaces," *IEEE Trans. Consum. Electron.*, vol. 56, no. 3, pp. 1364–1372, Aug. 2010.
[184] G. Schiavone, E. Guglielmelli, F. Keller, L. Zollo, and F. Chersi, "A wearable ergonomic gaze-tracker for infants," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, vol. 10. Aug. 2010, pp. 1283–1286.
[185] R. Mantiuk, M. Kowalik, A. Nowosielski, and B. Bazyluk, "Do-it-yourself eye tracker: Low-cost pupil-based eye tracker for computer graphics applications," in *Advances in Multimedia Modeling* (Lect. Notes Comput. Sci.), vol. 7131, K. Schoeffmann, B. Merialdo, A. G. Hauptmann, C.-W. Ngo, Y. Andreopoulos, and C. Breiteneder, Eds. Berlin, Germany: Springer-Verlag, 2012, pp. 115–125.
[186] H. Murphy and A. T. Duchowski, "Gaze-contingent level of detail," in *Eurographics*, J. Roberts, Ed. Manchester, U.K.: Univ. Manchester, 2001, pp. 219–228.
[187] A. T. Duchowski, E. Medlin, N. Cournia, A. Gramopadhye, B. Melloy, and S. Nair, "3D eye movement analysis for VR visual inspection training," in *Proc. Symp. Eye Tracking Res. Appl.*, vol. 2. 2002, p. 103.
[188] S. S. Mozafari Chanijani, S. S. Bukhari, and A. Dengel, "Analysis of text layout quality using wearable eye trackers," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, Turin, Italy, 2015, pp. 1–6.
[189] C. Topai, A. Dogan, and N. Gerek, "A wearable head-mounted sensor-based apparatus for eye tracking applications," in *Proc. IEEE Conf. Virtual Environ., Hum.-Comput. Interfaces Meas. Syst.*, Jul. 2008, pp. 136–139.
[190] Y. Itoh and G. Klinker, "Interaction-free calibration for optical see-through head-mounted displays based on 3D Eye localization," in *Proc. IEEE Symp. 3D User Interfaces*, Minneapolis, MN, USA, Mar. 2014, pp. 75–82.
[191] A. Steptoe *et al.*, "Eye tracking for avatar eye gaze control during object-focused multiparty interaction in immersive collaborative virtual environments," in *Proc. IEEE Virtual Reality Conf.*, Mar. 2009, pp. 83–90.
[192] M. Köles and K. Hercegfi, "Eye tracking precision in a virtual CAVE environment," in *Proc. 6th IEEE Int. Conf. Cogn. Infocommun.*, Oct. 2015, pp. 319–322.
[193] R. G. Bozomitu, V. Cehan, R. G. Lupu, C. Rotariu, and C. Barabaşa, "A new technique for improving pupil detection algorithm," in *Proc. Int. Symp. Signals, Circuits Syst.*, Iasi, Romania, Jul. 2015, pp. 1–4.
[194] T. Kocejko, J. Ruminski, J. Wtorek, and B. Martin, "Eye tracking within near-to-eye display," in *Proc. 8th Int. Conf. Hum. Syst. Interact.*, 2015, pp. 166–172.
[195] K. A. I. Essig, M. Pomplun, and H. Ritter, "A neural network for 3D gaze recording with binocular eye trackers," *Int. J. Parallel, Emergent Distrib. Syst.*, vol. 21, no. 2, pp. 79–95, Apr. 2006.
[196] S. Hillaire, A. Lecuyer, R. Cozot, and G. Casiez, "Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments," in *Proc. IEEE Virtual Reality Conf.*, Reno, NE, USA, Mar. 2008, pp. 47–50.
[197] V. Tanriverdi and R. J. K. Jacob, "Interacting with eye movements in virtual environments," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, New York, NY, USA, 2000, pp. 265–272.
[198] A. Plopski, Y. Itoh, C. Nitschke, K. Kiyokawa, G. Klinker, and H. Takemura, "Corneal-imaging calibration for optical see-through head-mounted displays," *IEEE Trans. Vis. Comput. Graphics*, vol. 21, no. 4, pp. 481–490, Apr. 2015.
[199] J. Turner, A. Bulling, and H. Gellersen, "Extending the visual field of a head-mounted eye tracker for pervasive eye-based interaction," in *Proc. Symp. Eye Tracking Res. Appl.*, vol. 1. 2012, pp. 269–272.
[200] Y. Kim and S. Jo, "Wearable hybrid brain-computer interface for daily life application," in *Proc. 3rd Int. Winter Conf. Brain-Comput. Interface*, Sabuk, South Korea, 2015, pp. 1–4.
[201] X. Fu, Y. Zang, and H. Liu, "A real-time video-based eye tracking approach for driver attention study," *Comput. Informat.*, vol. 31, no. 4, pp. 805–825, 2012.
[202] T. Poitschke, F. Laquai, S. Stamboliev, and G. Rigoll, "Gaze-based interaction on multiple displays in an automotive environment," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2011, pp. 543–548.
[203] A. Doshi and M. Trivedi, "A comparative exploration of eye gaze and head motion cues for lane change intent prediction," in *Proc. IEEE Intell. Vehicles Symp.*, Eindhoven, the Netherland, Jun. 2008, pp. 49–54.
[204] M. Sodhi, B. Reimer, and I. Llamazares, *Behavior Research Methods, Instruments, & Computers*, vol. 34. New York, NY, USA: Springer-Verlag, 2002, p. 529.
[205] Y.-L. Chen *et al.*, "Real-time eye detection and event identification for human-computer interactive control for driver assistance," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2014, pp. 2144–2149.
[206] D. Kern, A. Mahr, S. Castronovo, A. Schmidt, and C. Müller, "Making use of drivers' glances onto the screen for explicit gaze-based interaction," in *Proc. 2nd Int. Conf. Autom. User Interfaces Interact. Veh. Appl.*, 2010, p. 110.
[207] T. W. Victor, J. L. Harbluk, and J. A. Engström, "Sensitivity of eye-movement measures to in-vehicle task difficulty," *Transp. Res. F, Traffic Psychol. Behaviour*, vol. 8, no. 2, pp. 167–190, Mar. 2005.
[208] Z. Wuhe, Z. Lei, and D. Ning, "Sensing driver awareness by combining fisheye camera and Kinect," *Proc. SPIE*, vol. 9276, p. 927624, Nov. 2014.
[209] S. Han, S. Yang, J. Kim, and M. Gerla, "EyeGuardian: A framework of eye tracking and blink detection for mobile device users," in *Proc. 12th Workshop Mobile Comput. Syst. Appl.*, New York, NY, USA, 2012, Art. no. 6.
[210] Z. Li, G. Sun, F. Zhang, L. Jia, K. Zheng, and D. Zhao, "Smartphone-based fatigue detection system using progressive locating method," *IET Intell. Transp. Syst.*, vol. 10, no. 3, pp. 148–156, 2016.
[211] R. Biedert, A. Dengel, G. Buscher, and A. Vartan, "Reading and estimating gaze on smart phones," in *Proc. Symp. Eye Tracking Res. Appl.*, New York, NY, USA, 2012, pp. 385–388.
[212] S. Cheng, "The research framework of eye-tracking based mobile device usability evaluation," in *Proc. 1st Int. Workshop Pervasive Eye Tracking Mobile Eye-Based Interact.*, New York, NY, USA, 2011, pp. 21–26.
[213] A. Lopez-basterretxea and A. Mendez-zorrilla, "Eye/head tracking technology to improve HCI with iPad applications," *Sensors*, vol. 15, no. 2, pp. 2244–2264, 2015.
[214] J. V. Singh and G. Prasad, "Enhancing an eye-tracker based human-computer interface with multi-modal accessibility applied for text entry," *Int. J. Comput. Appl.*, vol. 130, no. 16, pp. 16–22, 2015.

ANURADHA KAR is currently pursuing the Ph.D. degree with the National University of Ireland Galway and the Center for Cognitive, Connected, and Computational Imaging. Her research interests include human computer interaction and computational imaging. She is involved in eye gaze tracking-addressing the issues of accuracy and performance evaluation of gaze estimation systems in various platforms.

PETER CORCORAN (F'10) is currently a Professor and a Personal Chair with the College of Engineering and Informatics, NUI Galway. In addition to his academic career, he is also an Occasional Entrepreneur, an Industry Consultant, and a Compulsive Inventor. He has co-authored over 300 technical publications and co-inventor on more than 250 granted U.S. patents. His research interests include biometrics, cryptography, computational imaging, and consumer electronics. He is the Editor-in-Chief and the Founding Editor of the *IEEE Consumer Electronics Magazine*.

Appendix B: Eye-Gaze Systems – An Analysis of Error Sources and Potential Accuracy in Consumer Electronics Use Cases

Eye-Gaze Systems – An Analysis of Error Sources and Potential Accuracy in Consumer Electronics Use Cases

Anuradha KAR, Shabab BAZRAFKAN, Claudia COSTACHE, and Peter CORCORAN, *Fellow, IEEE*
Center for Cognitive, Connected & Computational Imaging, College of Engineering & Informatics,
NUI Galway, Galway, Ireland
Email: {a.kar2, s.bazrafkan1, claudia.iancucostache, peter.corcoran}@nuigalway.ie

***Abstract*— Several generic CE use cases and corresponding techniques for eye gaze estimation (EGE) are reviewed. The optimal approaches for each use case are determined from a review of recent literature. In addition, the most probable error sources for EGE are determined and the impact of these error sources is quantified. A discussion and analysis of the research outcome is given and future work outlined.**

Keywords: eye gaze estimation methods, gaze estimation error sources, eye-trackers, smartphones.

I. INTRODUCTION

In the consumer electronics sector, research on eye gaze estimation techniques and applications have so far been focused on static setups- e.g. desktop computer, TV panels [4], or gaming applications [5]. However, with the broad adoption of smartphone devices the consumer electronics industry has recently begun to focus on the use of eye gaze for HCI in dynamic platforms [6]. Commercial adaptation of gaze tracking in handheld devices is extremely challenging and has been less than successful so far, because of poor accuracy, reliability and usability issues.

In this paper we examine and broadly quantify a range of error sources that affect eye gaze estimation in consumer devices. Two main use cases are considered – the static case for TV or game consoles and the dynamic use case for handheld devices or automotive applications. It is shown that the cumulative errors are significant and call into question the practicality of eye gaze as means of HCI in many use cases. The review and analysis of the algorithms and error sources will be of interest to engineers and researchers working in this field and should provide a useful reference when defining design criteria for specific consumer electronics use cases.

II. EYE GAZE ESTIMATION APPROACHES

A. Eye gaze estimation methods

There are two main approaches for video-based EGE:

1) Appearance based methods

Appearance based methods [6] detect and track eye gaze based on the photometric appearance (color distribution or filter responses) of the eye-region, using images or video sequences. The main advantage is simple hardware available in most devices (e.g. laptops, tablets, and smartphones) whereas the downside is low accuracy, requirement of large training datasets and sensitivity to head pose variations [2].

2) Feature based methods

These require more sophisticated hardware such as one or more near infrared (NIR) cameras and NIR LEDs and use the LED glints on the cornea and pupil to estimate the eye gaze [1]. There are two main approaches [3]: (i) *regression based* methods that map image features to gaze co-ordinates in either polynomial or non-parametric forms; (ii) *model based* methods that use a geometrical model of the eye to estimate gaze direction vector and its intersection with scene geometry.

B. Eye gaze use cases

Based on potential error sources, any EGE problem can be associated with three main types of use cases.

1) Head mounted eye gaze systems

This EGE scheme (used in gaming and virtual reality applications) requires that the user wears a head mounted device so that the camera is in a fixed position relative to the user eye and can detect gaze points with high accuracy [10].

2) Fixed gaze tracking systems

These systems use a fixed platform to position the camera(s) and LED(s) and use either appearance or feature-based methods. These systems are currently used in IPTVs, desktop based and driver-assist systems [3-4]. These systems can compensate for minor changes in head poses, but the overall accuracy is lower as described in section III.

3) Mobile gaze tracking systems

Design and implementation of EGE on mobile devices is a non-trivial task. Information on device motion and user head position is needed and several unique error sources combine to significantly affect the tracking accuracy and resolution [5].

III. SOURCES OF ERROR

A. Head pose changes

Head pose variations lead to failure in eye gaze tracking by changing in eye-socket geometry and to the user field of view or causing the LED glints in the cornea- surface to disappear [8]. Table 1 shows the tolerance of head pose variations for various EGE techniques.

B. Hand movements – position, orientation and jitter

Additional error sources, especially for handheld devices are changes in hand pose, i.e., the way a user holds the device and hand- jitter (low frequency hand vibrations) which vary according to user habits and physical characteristics.

The research work presented here was funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded FotoNation Ltd. Project ID: 13/SPP/I2868 on "*Next Generation Imaging for Smartphone and Embedded Platforms*".

978-1-4673-8364-6/16/$31.00 ©2016 IEEE

Table 1: Accuracy under head pose variations in various EGE methods

Method	Platform	Accuracy
Appearance based [7]	Desktop, webcam(1280x1024)	1deg(fixed head) 2deg(slight head pose)
Feature(model)based [12]	2 cameras ring of LEDs for each camera	0.7deg(fixed head) 1.5deg(slight head motion)
Feature(regression)based [9]	1 camera 1NIR source	0.4 deg(fixed head) 1 deg(slight head pose)

C. *Eye socket resolution*

The pixel resolution of the eye image captured by the EGE camera will define the gaze accuracy that can be achieved. Factors such as camera resolution, image contrast, distance of the device from the eye affects the amount of eye details in the captured image and hence the accuracy of gaze tracking. For example, the typical pixel resolution for the iris region in images taken by a 5 MP front camera of a smartphone, at arm distance (45 cm), with indoor illumination is around 30 pixels.

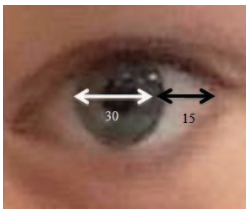


Fig. 1. Pixel resolution of eye images taken with a smartphone camera

D. *Human eye limitations*

Human eyes can fixate as accurately as 10 minutes of visual angle (0.16 deg) and therefore theoretically EGE systems cannot exceed this accuracy limit. Also high frequency eye movements or saccades lead to motion blur and missing gaze points if frame-rate of EGE camera is lower than 100 Hz [11].

E. *Changes in illumination*

Unpredictable changes in illumination will affect eye feature detection, and may cause additional glints to appear on the eye cornea surface, disturbing the feature based algorithms [8].

F. *Other sources of error*

These include eye occlusion from wearing glasses, facial expressions and skin color, which introduce errors that propagate throughout the gaze estimation procedure. For each use case, combinations of these error sources may exist.

Table 2 shows the error sources encountered for each use case. It is seen that handheld devices experience a wider range of error sources compared to head mounted systems and present a much more challenging problem to realize EGE.

Table 2: Existence of error for different use cases

Error source\use cases	Head mounted	Fixed	Mobile
Eye socket resolution	x	x	x
Head pose	--	x	x
Algorithm error	x	x	x
Hand jitter	--	--	x
Hand pose	--	--	x
Change in illumination	--	x	x
Human limitation	x	x	x
Other sources	--	x	x

IV. Experimental Setup

Table 3: Eye-tracker specifications

Parameters	Values
Accuracy	0.5° – 1°
Horizontal range	45cm – 75cm
Sampling Frequency	30Hz / 60Hz
Calibration	9, 12 or 16 points
Latency	<20ms at 60Hz

Fig.2 A portable eye tracker for EGE studies and specifications

Our setup includes a portable eye-tracker mountable on desktop computers or tablets, for studying the errors in static and dynamic EGE use cases, along with the front cameras and motion sensors of several smartphone models.

V. Conclusion

For fixed EGE setups, some of the error sources have been quantified, but there is limited information about such error estimates for devices operating under dynamic conditions. As some of these error sources are not completely independent (e.g. eye socket resolution- a function of camera resolution and distance from eye to device, is also dependent on hand and head movements), it is not possible to determine the total error by a linear combination of individual errors.

Additional details on specific use-cases of EGE will be presented at the ICCE including a detailed discussion on the non-linear nature of some error sources based on CE devices, with design guidance for engineers seeking to introduce real-time eye-gaze as HCI component in these platforms.

References

[1] E. D. Guestrin and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1124–1133, 2006.

[2] D. W. Hansen and Q. Ji, "In the eye of the beholder: a survey of models for eyes and gaze.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, 2010.

[3] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, "Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[4] P. M. Corcoran, F. Nanu, S. Petrescu, and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," *Consum. Electron. IEEE Trans.*, vol. 58, no. 2, pp. 347–355, 2012.

[5] P. Corcoran, "To Gaze with Undimmed Eyes on All Darkness [IP Corner]," *IEEE Consum. Electron. Mag.*, vol. 4, no.1, pp 99–103, 2015.

[6] Y. Sugano, Y. Matsushita, and Y. Sato, "Appearance-based gaze estimation using visual saliency," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 35, no. 2, pp. 329–341, 2013.

[7] A. Al-Rahayfeh and M. Faezipour, "Application of head flexion detection for enhancing eye gaze direction classification," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, 2014, pp. 966–969.

[8] R. Valenti and T. Gevers. Accurate eye center location and tracking using isophote curvature. In CVPR, pages 1-8, 2008.

[9] L. Sesma-Sanchez, A. Villanueva, and R. Cabeza, "Gaze estimation interpolation methods based on binocular data," *Biomed. Eng. IEEE Trans.*, vol. 59, no. 8, pp. 2235–2243, 2012.

[10] Hammoud R et al, Passive Eye Monitoring - Algorithms, Applications and Experiments, Springer, Berlin, 2008

[11] W. W. Abbott and A. A. Faisal "Ultra-low-cost 3D gaze estimation:An intuitive high information throughput compliment to direct brain-machineinterfaces", J. Neural Eng., vol. 9, no. 4, pp.046016 2012.

[12] C.-C. Lai, S.-W. Shih, and Y.-P. Hung, "Hybrid Method for 3-D Gaze Tracking Using Glint and Contour Features," *Circuits Syst. Video Technol. IEEE Trans.*, vol. 25, no. 1, pp. 24–37, 2015

978-1-4673-8364-6/16/$31.00 ©2016 IEEE

Appendix C: Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms

Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms

Anuradha Kar*, Peter Corcoran*
* Center for Cognitive, Connected & Computational Imaging, College of Engineering & Informatics, NUI Galway, Galway, Ireland
Email: {a.kar2, peter.corcoran}@nuigalway.ie

Abstract— **There is a need to standardize the performance of eye gaze estimation (EGE) methods in various platforms for human computer interaction (HCI). Because of lack of consistent schemes or protocols for summative evaluation of EGE systems, performance results in this field can neither be compared nor reproduced with any consistency. In contemporary literature, gaze tracking accuracy is measured under non-identical sets of conditions, with variable metrics and most results do not report the impact of system meta-parameters that significantly affect tracking performances. In this work, the diverse nature of these research outcomes and system parameters which affect gaze tracking in different platforms is investigated and their error contributions are estimated quantitatively. Then the concept and development of a performance evaluation framework is proposed- that can define design criteria and benchmark quality measures for the eye gaze research community.**

I. INTRODUCTION

The prolific and interdisciplinary nature of research on eye gaze estimation in the past decades has resulted in the development of a wide range of techniques and applications [1-4]. Most of the recent developments in this field comprise of passive video-based gaze tracking methods which estimate the gaze direction or the point of gaze by capturing and processing images of the eye region in natural light or under active near infrared (NIR) illumination. Eye gaze based applications are also implemented in a variety of consumer platforms such as desktop setups (for text entry, attention analysis, gaze based passwords), head-mounted equipment (gaming, virtual and augmented reality), automotive systems (driver alertness monitoring) and handheld devices (gaze based scrolling, content navigation) [5-15].

In spite of the large volume of research on EGE techniques over the years, currently there are no unified standard schemes to evaluate the performance of gaze tracking algorithms or setups in research across various platforms. Few works report the impact of system parameters such as camera and screen resolution, viewing angle and platform movements- that play significant roles in determining the ultimate accuracy of a gaze tracking system. Also there is no agreement on the metrics of accuracy across various methods. Therefore it is difficult to state with certainty if the results of a recent design would perform better than conventional ones and under what conditions certain research claims are valid.

The aim of the work presented in this paper is to address the challenges of standardization in the development and performance evaluation of gaze based HCI systems. First a literature review is provided where it is described how the implementation and outcomes in eye gaze research differ-to the extent that they are neither comparable nor reproducible. Then several sources of error including head pose variations, viewing angle, camera resolution are identified and their impact is quantified experimentally. Finally the development of a framework for evaluating the performance of EGE setups is outlined. Such a concept can be used to provide reliable estimates of error and standardized accuracy scores to users and designers of gaze based systems.

II. LITERATURE REVIEW

A comprehensive literature review on the development of EGE techniques and applications for different consumer platforms was made and the diversity in the algorithms, research outcomes, setups, sources of errors and accuracy measures for different platforms is presented below.

A. Eye gaze tracking algorithms in literature

These fall broadly into the following classes [1]: 1) Feature based: These use single or multiple near infrared (NIR) cameras and LEDs to produce glints on the cornea and use the vector between the pupil center and glint positions to track the eye gaze. These are further subdivided as: (i) regression based methods that use a polynomial regression function to map the vector to gaze coordinates on screen (ii) 3D model based methods that use a geometrical model of the eye to develop accurate ray tracing to estimate gaze direction. Feature based methods typically have good accuracy (0.5 to 1 degree) but need elaborate hardware. 2) Appearance based: These utilize the shape and appearance features of the eye region to train a model which is then used to match with captured eye images for estimating gaze. These methods need simple hardware but have lower accuracy [40].

B. Eye gaze use cases in consumer platforms

Use of eye gaze information has found applications in a variety of user platforms –some of them are described along with their recent and popular applications.

978-1-5090-1897-0/16/$31.00 ©2016 IEEE

SMC_2016 002061

a) Desktop based: Majority of eye gaze tracking methods have been developed for desktop based systems [30-32] wherein one or more cameras with NIR LEDs are used to track a user's eye gaze on the computer screen-with the user seated in front of the computer. Some methods allow free head movement whereas some require a fixed user head position for accurate tracking. Typical tracking methods- like PCCR (pupil center corneal reflection) or appearance models are used for gaze based applications ranging from assistive technologies for the physically impaired, understanding the development of psychiatric disorders, e-learning, studying consumer attention patterns, e-commerce and web-design [2].
b) TV panels: Gaze controlled intelligent TVs have recently been introduced [33-36] that uses PCCR techniques for gaze localization. In these systems, a user can select and navigate menus and switch channels by looking at icons shown on the TV's display.
c) Automotive: Interactive driver support systems have been developed based on actively tracking driver gaze or blinking patterns to evaluate driver vigilance and drowsiness levels [37-39]. The setups employ machine vision algorithms, appearance models or PCCR techniques using cameras and NIR light sources mounted on the car's dashboard.
d) Handheld devices: Methods for detecting user gaze location and patterns have been developed for smartphone and tablets [40-42] to activate functions such as locking and unlocking devices, interactive displays, dimming backlights or suspending sensors based on user attention. The operating principles rely on PCCR methods, appearance models or ellipse fitting techniques using IR light sources and the device front camera.
e) Head-mounted setups: These usually comprise of two or more cameras mounted on a head unit – one facing the user and recording the eye gaze (called the 'eye camera'), while the other facing outward, recording the scene (called 'scene camera'). These setups allow free head movements. Major applications include virtual and augmented reality, observing user attention, psychoanalysis and occulo-motor movements [43-45]. In Table I the characteristics of setups for gaze estimation in various use cases are summarized.

TABLE I
FEATURES OF EGE SYSTEMS IN VARIOUS USE CASES

EGE use cases	Distance between eye and tracker	Viewing angle (deg)	Screen size (inch)
Desktop	30-50 cm	~40	14-17
TV panels	2-5 m	60-120	26-70
Automotive	50 cm	40-60	--
Handheld devices	20-40 cm	5-12	5-10
Head mounted	2-5 cm	55-75	--

C. Sources of errors in EGE systems

A variety of error sources affect gaze tracking accuracy in different platforms. These include: a) Head-pose changes- which alters the eye-socket geometry, user field of view, causes LED glints on cornea-surface to disappear [8]. b) Camera resolution- affects amount of eye details and contrast in the eye image captured by the EGE setup c) Display properties and viewing angle-includes effect of size and pixel resolution of the screen where gaze is tracked and distance of the user from setup d) Platform movements–may cause variable position, orientation and jitter in the setup e) Changes in illumination- affects eye feature detection, causes additional glints to appear on cornea surface. f) Human eye limitations-the eyes can fixate as accurately as 10 minutes of visual angle (0.16 degree) which sets the accuracy limit of gaze tracking. High frequency eye movements or saccades lead to motion blur and missing gaze points if the frame-rate of the gaze estimation camera is less than 100 Hz [29].
However the problem is that only a few of these factors are reported in literature while the impact of the others is not considered at all. As an illustrative example, the effect of head pose variations are considered in [8-10] whereas impact of factors such as camera quality, display size and resolution, user viewing angle and platform movements are inadequately characterized and therefore it is not possible to know as to how an algorithm would perform under their influence. The occurrence of the various error sources in different user platforms are tabulated below.

TABLE II
SOURCES OF ERRORS IN EGE SYSTEMS

Error Sources	Head Mounted	Desktop	Dynamic
Head pose	--	X	X
Camera resolution	X	X	X
Display properties	--	X	X
Viewing angle	--	X	X
Platform motion	--	--	X
Illumination changes	--	X	X
Human eye limitation	X	X	X

D. Diversity in research outcomes in EGE literature

The metrics used to represent performance scores of gaze estimation algorithms for different CE platforms were studied and it was found that a large number of reported results cannot be compared in any meaningful way as the system performance is reported in varied formats. Some papers [16-20] report tracking accuracy in degrees while others report them as gaze recognition rates [21-24]. Again there are results where accuracy is reported in heterogeneous formats-e.g. relative pixel/distance shifts or rate of correct detections which have relevance solely to the procedure stated in the paper [25-28]. Apart from this, wide variations in setup configurations are noted in terms of the number of cameras and light sources used-varying between 1 to 4 cameras and 1 to 16 LEDs. It is also observed that head mounted setups appear to have a lower mean error (~1.08 degree) compared to desktop based (~1.87 degree) or dynamic platforms (~4.8 degree). However each platform is affected by several error sources and unique operating conditions as described in the previous sections, and therefore the absolute validity of these results are uncertain.

SMC_2016 002062

III. QUANTIFYING EFFECTS OF ERROR SOURCES IN EGE

A set of eye tracking experiments were conducted to simulate and observe the impact of various error sources mentioned above. These include studying head pose tolerance levels and effects of viewing angle, display characteristics and camera resolution on gaze tracking accuracy. For the setup a commercial eye tracker that works with a laptop or desktop system and has a specified gaze tracking accuracy of 0.5 degree was used. Eye tracking data is collected from users who were asked to sit in front of the eye tracker connected to a computer as shown in Fig. 1 and an eye calibration routine for the user is run first. The experimental flowchart is shown in Fig3 where each experiment in general comprises of recording the eye gaze coordinates from the tracker when the user gazes and clicks on fixed points on the screen. The gaze error in degrees is calculated from the shift between click and gaze locations on the screen as shown in Fig 2 Further details of the setup and experiments are described below.

A. Head pose tolerance

In this experiment a user is asked to sit in front of the tracker with their head initially positioned frontally. A video camera is used to capture the position of the head and the head pose in roll, pitch and yaw angles is obtained from an appearance model as shown in Fig 4. For each position gaze accuracy data is recorded. Then the user is asked to turn their head to specific fixed positions (in roll pitch yaw angles) and perform the gaze tracking experiment described above. Gaze accuracy measures corresponding to various head pose angles are recorded. The results are presented in Fig.5 below. It is observed that the gaze tracking accuracy significantly decreases as the head moves slightly beyond 20 degrees in either directions of roll, pitch and yaw angles.

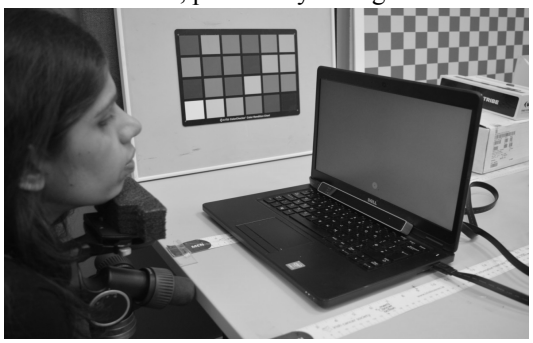


Fig.1 Experimental setup with the user, laptop and eye-tracker with the user head fixed with a chin-rest

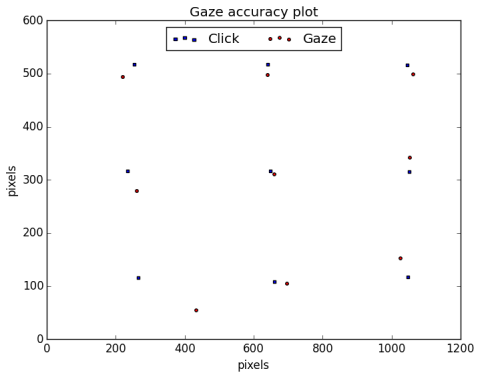


Fig2. Output from an eye tracking experiment

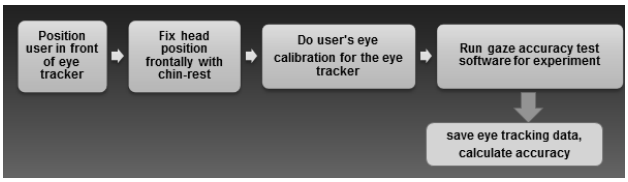
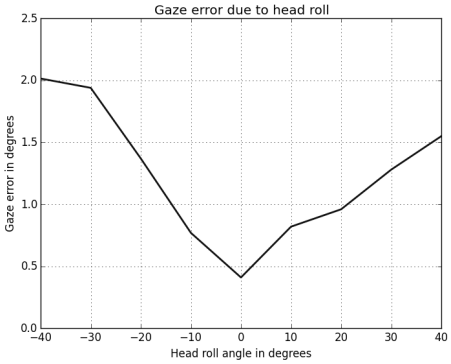


Fig.3 Experimental flowchart

Fig.4 Estimating head pose angles using a face model



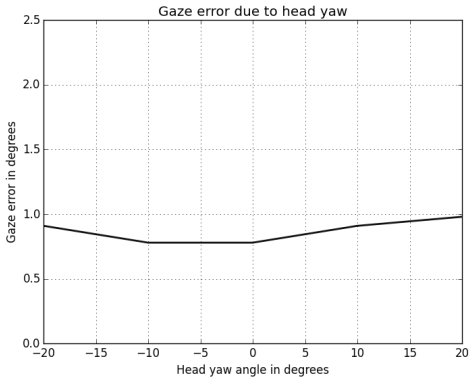


Fig5 Variation of gaze angular error with head pose angles

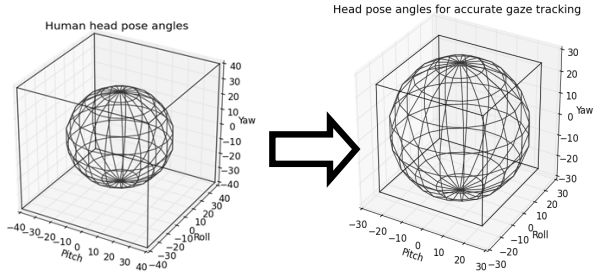


Fig6 Head pose tolerance limits of the eye tracker

The human head has considerable degree of flexibility in the order of nearly 40 degrees of angular movement. However the above results imply that for reliable eye tracking results with the given tracker, the user needs to keep their head position limited within an angular "box"-equivalent to ~20 degrees about the central position in 3 directions as shown in Fig. 6 or the gaze tracking performance will drop below acceptable levels.

B. Viewing angle

For this experiment, the users were positioned at successively increasing distances from the tracker - computer screen setup (40 cm to 100 cm in 10 cm interval) and the gaze tracking accuracy data were recorded for each user position for fixed frontal head poses and plotted in Fig 7 below. The user viewing angle decreases as the user moves away from the screen as shown in Fig. 7(top). It is observed that for the tracker, the accuracy decreases rapidly with an increased viewing angle. However it is also found that the tracker errors are high when the user is too close to the tracker-that is the user to tracker-screen setup is less than 40 cm.

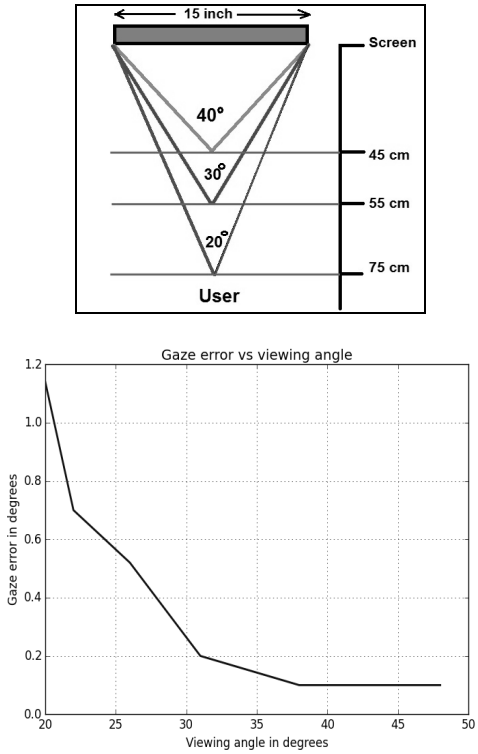


Fig7. Top: Variation of user viewing angle with distance from screen. Bottom: Variation of gaze accuracy as a function of user distance.

C. Display size and resolution

The eye tracking experiment was run on displays of various sizes (9, 11, 13.5, 15 inches) and resolutions (800x600, 1024x768, 1280x768, 1366x768) and the corresponding gaze tracking errors are shown below.

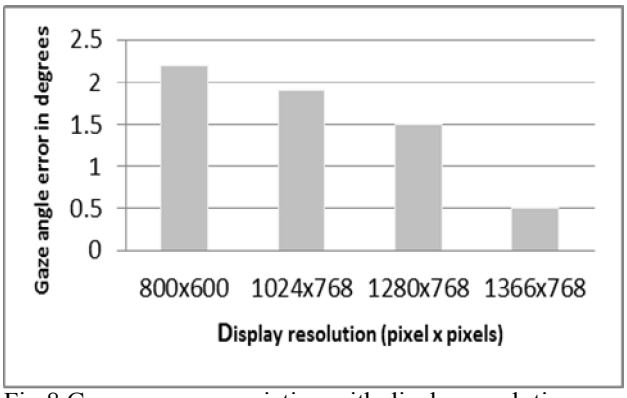


Fig.8 Gaze accuracy variation with display resolution

D. Camera resolution

Every gaze tracking device has one or more cameras to capture images of the eye region and the camera resolution directly affects the amount of eye details in the captured image and hence the accuracy of gaze tracking [29].To study the effect of camera resolution on gaze estimation errors, eye images were captured using camera resolutions starting from 1.3 to 24 megapixels for different distances between the camera and user under constant illumination as shown in Fig.9.

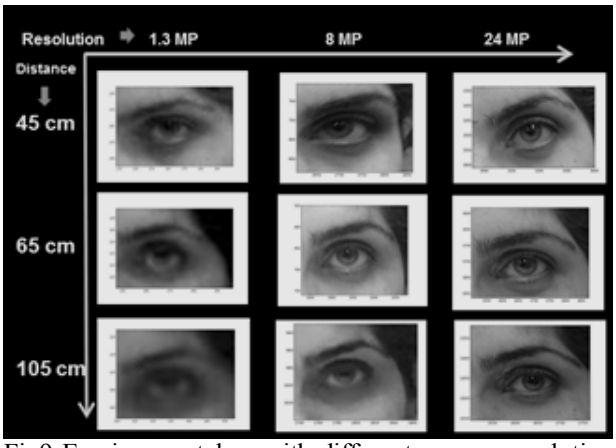


Fig9 Eye images taken with different camera resolutions at different distances from the user

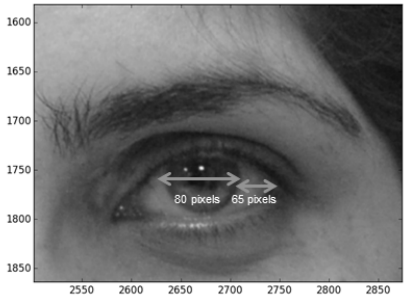


Fig10. Details of the eye socket region in an eye image

The eye details in an image are estimated as the eye socket width in pixels as shown in Fig. 10 above. The eye-details as a function of distance from the camera are estimated and plotted in Fig.11 & 12. It is observed that for a camera with low resolution the eye details are inherently low and the image quality reduces drastically as the eyes move away from the camera. With a high quality camera the eye details remains nearly constant with distance. To observe how the camera quality might affect eye gaze tracking- a pupil detection algorithm on cropped eye images was applied using circular Hough transform (Fig.13). The shift in positions between the actual and detected pupil centers is estimated as pupil detection errors. The plot of errors as a function of camera resolution is shown in Fig14. It is seen that pupil detection errors reduce with better camera quality-but levels off around 8 MP which could be an optimal camera resolution that can be used to build reliable remote eye trackers operating in the horizontal range of 1meter.

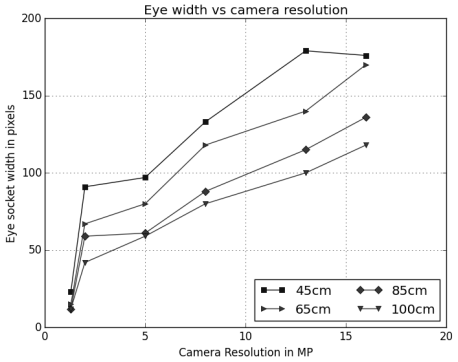


Fig11 Variation of eye details with camera resolution.

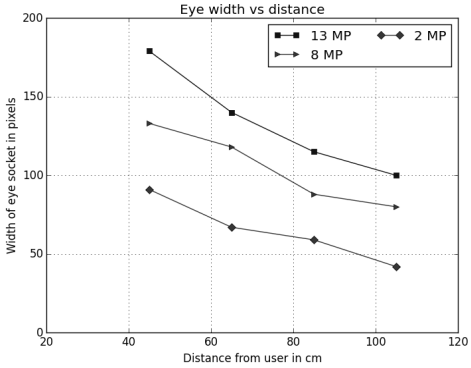


Fig12 Variation of eye details with varying distance between camera and user for different camera resolutions

IV. Discussions and Future Work

In this paper, the diverse range of algorithms, performance metrics and system specifications of gaze based HCI systems was investigated. Then a set of experiments were designed and implemented to estimate the impact of various sources of errors that affect EGE performance. The major contribution of our study was identifying two main issues that affect realistic performance evaluation of conventional gaze based systems. Firstly there is a lack of protocols for reporting research outcomes and accuracy scores in literature.

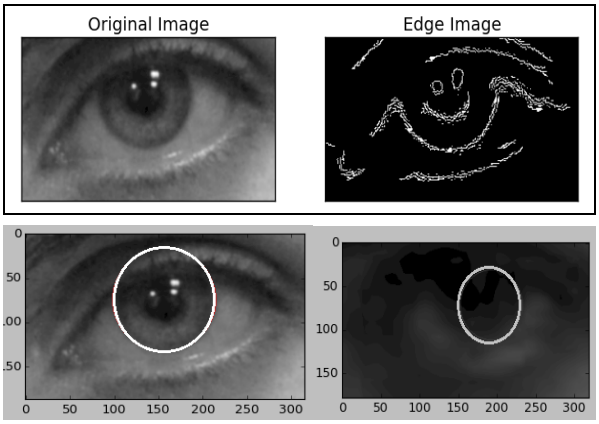


Fig.13 Top: Applying pupil detection on eye images. Bottom: Pupil detection in good and poor quality images

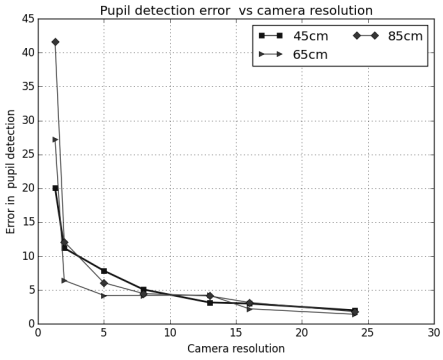


Fig14. Pupil detection errors as a function of camera resolution and distance from the user.

Secondly, there are several parameters in EGE system development that potentially affect tracking performance, but their impacts are very sparsely evaluated and reported by researchers in this field. Our studies on various error sources showed that error contribution from these parameters are significant and it can be concluded that the reported results on gaze tracking techniques that are measured under controlled setup conditions do not give a correct estimate about the overall accuracy of the system.

With this background we are designing a unified framework that would provide standardized performance scores of gaze tracking systems in different platforms while taking into consideration the multiple error sources that may affect such a system. The framework is expected to operate through multiple control experiments wherein each error source is evaluated separately. It may be extended to estimate the summative error in the system as a measure of its true performance. Through such a framework it will be possible to evaluate and compare the performance of multiple algorithms while measuring specific gaze behavior/actions (e.g. pursuit/fixation), under uniform experimental conditions and identify performance bottlenecks of different algorithms when they perform under identical circumstances. The ultimate aim is to provide more reliable performance measures and design guidelines that will facilitate both the future users and designers of advanced EGE systems.

ACKNOWLEDGMENT

The research work presented here was funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded FotoNation Ltd. Project ID: 13/SPP/I2868 on "*Next Generation Imaging for Smartphone and Embedded Platforms*".

REFERENCES

[1] D. W. Hansen and Q. Ji, "In the eye of the beholder: a survey of models for eyes and gaze.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, 2010.

[2] C. H. Morimoto and M. R. M. Mimica, "Eye gaze tracking techniques for interactive applications," *Comput. Vis. Image Underst.*, vol. 98, no. 1, pp. 4–24, 2005.

[3] G. Vasilescu, "Springer Series on Signals and Communication Technology Signals and Communication Technology," *Electron. Noise Interf. Signals*, 2005.

[4] F. Song, X. Tan, S. Chen, and Z. H. Zhou, "A literature survey on robust and efficient eye localization in real-life scenarios," *Pattern Recognit.*, vol. 46, no. 12, pp. 3157–3173, 2013.

[5] P. M. Corcoran, F. Nanu, S. Petrescu and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," in *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 347-355, May 2012

[6] A. Bulling, F. Alt, and A. Schmidt, "Increasing the security of gaze-based cued-recall graphical passwords using saliency masks," *Proc. 2012 ACM Annu. Conf. Hum. Factors Comput. Syst. - CHI '12*, p. 3011, 2012.

[7] S. H. Lee, J. Y. Lee, and J. S. Choi, "Design and implementation of an interactive HMD for wearable AR system," *2011 17th Korea-Japan Jt. Work. Front. Comput. Vision, FCV 2011*, 2011.

[8] A. T. Duchowski, E. Medlin, N. Cournia, A. Gramopadhye, B. Melloy, and S. Nair, "3D eye movement analysis for VR visual inspection training," *Proc. Symp. Eye Track. Res. Appl. - ETRA '02*, p. 103, 2002.

[9] I. H. Choi and Y. G. Kim, "Head pose and gaze direction tracking for detecting a drowsy driver," *Appl. Math. Inf. Sci.*, vol. 9, no. 2, pp. 505–512, 2015.

[10] A. Doshi and M. M. Trivedi, "On the roles of eye gaze and head dynamics in predicting driver's intent to change lanes," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 453–462, 2009.

[11] H. Skovsgaard, J. Hansen, and E. Møllenbach, "Gaze Tracking Through Smartphones," *Work. Gaze Interact. Post-WIMP World CHI*, 2013.

[12] H. Drewes, A. De Luca, and A. Schmidt, "Eye-gaze interaction for mobile phones," *Mobil. '07 Proc. 4th Int. Conf. Mob. Technol. Appl. Syst. 1st Int. Symp. Comput. Hum. Interact. Mob. Technol.*, vol. 07, pp. 364–371, 2007.

[13] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, "Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[14] E. Wood and A. Bulling, "Eyetab: Model-based gaze estimation on unmodified tablet computers," *Etra*, pp. 3–6, 2014.

[15] M. F. Land, "Eye movements and the control of actions in everyday life," *Prog. Retin. Eye Res.*, vol. 25, no. 3, pp. 296–324, 2006.

[16] a Meyer, M. Böhme, T. Martinetz, and E. Barth, "A single-camera remote eye tracker," *Percept. Interact. Technol.*, vol. 4021, pp. 208–211, 2006.

[17] S. J. Baek, K. A. Choi, C. Ma, Y. H. Kim, and S. J. Ko, "Eyeball model-based iris center localization for visible image-based eye-gaze tracking systems," *IEEE Trans. Consum. Electron.*, vol. 59, no. 2, pp. 415–421, 2013.

[18] C. C. Lai, S. W. Shih, and Y. P. Hung, "Hybrid method for 3-D gaze tracking using glint and contour features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 1, pp. 24–37, 2015.

[19] A. Tsukada, M. Shino, M. Devyver, and T. Kanade, "Illumination-free gaze estimation method for first-person vision wearable device," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 2084–2091, 2011.

[20] Zhiwei Zhu and Qiang Ji, "Novel Eye Gaze Tracking Techniques Under Natural Head Movement," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 12, pp. 2246–2260, 2007.

[21] P. Smith, M. Shah, and N. da Vitoria Lobo, "Determining driver visual attention with one camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 4, pp. 205–218, 2003.

[22] L. M. Bergasa and J. Nuevo, "Real-time system for monitoring driver vigilance," *IEEE Int. Symp. Ind. Electron.*, vol. III, no. 1, pp. 1303–1308, 2005.

[23] Y. Liang, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 340–350, 2007.

[24] A. Tawari and M. M. Trivedi, "Robust and continuous estimation of driver gaze zone by dynamic analysis of multiple face videos," *IEEE Intell. Veh. Symp. Proc.*, no. Iv, pp. 344–349, 2014.

[25] C. Hennessey, B. Noureddin, and P. Lawrence, "A single camera eye-gaze tracking system with free head motion," *Measurement*, vol. 1, no. March, pp. 27–29, 2006.

[26] V. Vaitukaitis and A. Bulling, "Eye gesture recognition on portable devices," *Proc. 2012 ACM Conf. Ubiquitous Comput. - UbiComp '12*, p. 711, 2012.

[27] Y.-W. Sun, C.-K. Chiang, and S.-H. Lai, "Integrating eye tracking and motion sensor on mobile phone for interactive 3D display," vol. 8856, p. 88560F, 2013.

[28] T. Kocejko, J. Ruminski, J. Wtorek, and B. Martin, "Eye tracking within near-to-eye display," *Proc. - 2015 8th Int. Conf. Hum. Syst. Interact. HSI 2015*, no. 1, pp. 166–172, 2015.

[29] W. W. Abbott and a a Faisal, "Ultra-low-cost 3D gaze estimation: an intuitive high information throughput compliment to direct brain–machine interfaces," *J. Neural Eng.*, vol. 9, no. 4, p. 046016, 2012.

[30] D. LeBlanc, A. Forget, and R. Biddle, "Guessing click-based graphical passwords by eye tracking," *PST 2010 2010 8th Int. Conf. Privacy, Secur. Trust*, pp. 197–204, 2010.

[31] H. Drewes, R. Atterer, and A. Schmidt, "Detailed monitoring of user's gaze and interaction to improve future e-learning," *Univers. Access Hum. Comput. Interact. Ambient Interact. Proc. 4th Int. Conf. Univers. Access Hum. Comput. Interact. UAHCI2007, held as Part HCI Int. 2007*, pp. 802–811, 2007.

[32] S. Milekic, "Using Eye- and Gaze-tracking to Interact with a Visual Display," *Electron. Vis. Arts*, no. Yarbus, pp. 42–48, 2012.

[33] Y. L. Chen, C. Y. Chiang, C. W. Yu, W. C. Sun, and S. M. Yuan, "Real-time eye tracking and event identification techniques for smart TV applications," *Dig. Tech. Pap. - IEEE Int. Conf. Consum. Electron.*, pp. 63–64, 2014.

[34] D. T. Nguyen, K. Y. Shin, W. O. Lee, Y. G. Kim, K. W. Kim, H. G. Hong, K. R. Park, C. Oh, H. Lee, and Y. Jeong, "Gaze detection based on head pose estimation in smart TV," *Int. Conf. ICT Converg.*, pp. 283–288, 2013.

[35] D. M. Small, W. Y. Sanchez, M. J. Hickey, and G. C. Gobe, "Multiphoton fluorescence microscopy of the live kidney in health and disease Multiphoton fluorescence microscopy of the live kidney in health and disease," *J. Biomed. Opt.*, vol. 19, no. 2, p. 020901, 2014.

[36] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, "Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[37] J. H. Oh and N. Kwak, "Recognition of a Driver's gaze for vehicle headlamp control," *IEEE Trans. Veh. Technol.*, vol. 61, no. 5, pp. 2008–2017, 2012.

[38] P. Jiménez, L. M. Bergasa, J. Nuevo, N. Hernández, and I. G. Daza, "Gaze Fixation System for the Evaluation of Driver Distractions Induced by IVIS," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–12, 2012.

[39] X. Sun, L. Xu, and J. Yang, "Driver fatigue alarm based on eye detection and gaze estimation," *Proc. SPIE*, vol. 6786, no. 1, pp. 678612–678612–6, 2007.

[40] A. Kar, S. Bazrafkan, C. C. Ostache and P. Corcoran, "Eye-gaze systems - An analysis of error sources and potential accuracy in consumer electronics use cases," *2016 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, 2016, pp. 319-320.

Appendix D: Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations

sensors

MDPI

Article

Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations

Anuradha Kar * and Peter Corcoran

Department of Electrical & Electronic Engineering, National University of Ireland, Galway H91 TK33, Ireland; peter.corcoran@nuigalway.ie

* Correspondence: a.kar2@nuigalway.ie; Tel.: +353-834-386-560

Received: 10 August 2018; Accepted: 15 September 2018; Published: 18 September 2018

Abstract: An eye tracker's accuracy and system behavior play critical roles in determining the reliability and usability of eye gaze data obtained from them. However, in contemporary eye gaze research, there exists a lot of ambiguity in the definitions of gaze estimation accuracy parameters and lack of well-defined methods for evaluating the performance of eye tracking systems. In this paper, a set of fully defined evaluation metrics are therefore developed and presented for complete performance characterization of generic commercial eye trackers, when they operate under varying conditions on desktop or mobile platforms. In addition, some useful visualization methods are implemented, which will help in studying the performance and data quality of eye trackers irrespective of their design principles and application areas. Also the concept of a graphical user interface software named GazeVisual v1.1 is proposed that would integrate all these methods and enable general users to effortlessly access the described metrics, generate visualizations and extract valuable information from their own gaze datasets. We intend to present these tools as open resources in future to the eye gaze research community for use and further advancement, as a contribution towards standardization of gaze research outputs and analysis.

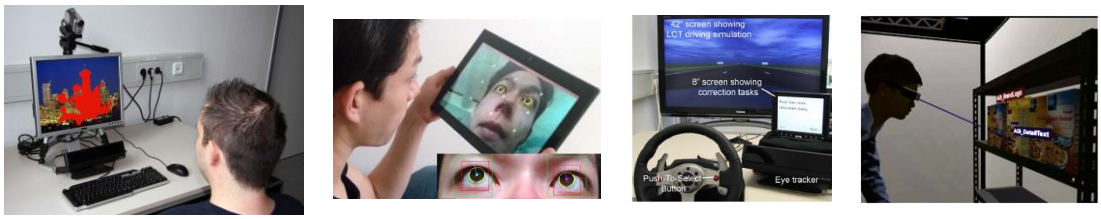
Keywords: eye gaze estimation; eye tracker; performance evaluation; metrics; visualizations; mobile devices; graphical user interface; open source; standardization

1. Introduction

Research works on eye gaze estimation typically present their results in a wide range of ways. While the commonly used measure of gaze tracking accuracy is angular resolution (in degrees), other metrics such as gaze recognition rates (in percentage) and shifts between estimated and target gaze locations (in pixels or mms) are used frequently. These metrics are not interrelated and sometimes not clearly defined, which leads to ambiguities in evaluating and comparing performances of gaze tracking systems. Table 1 below shows the statistics derived from a recent literature review done by the authors [1] surveying nearly 200 research articles on gaze based algorithms and applications, which highlights the current diversity in metrics used for representing gaze estimation performance. It may be observed from the table that although use of angular resolution as a metric is common, the uncorrelated metrics like percentage recognition rates and pixel distances are also used in many works. Column 2 of the table shows the total number of papers surveyed for each consumer platform or eye gaze use-case category. The other columns present the different metrics, with the number in each cell of them representing the number of papers where each type of metric is used. Figure 1 below shows illustrations of eye gaze applications in different consumer platforms like desktop, mobile devices, automotive and head mounted systems.

Table 1. Diversity in performance evaluation metrics used in eye gaze research articles.

Eye Tracking Platform	No. of Surveyed Papers	No of Papers with Metric: Degree	No. of Papers with Metric: Percentage	No. of Papers with Metric: Others (e.g., pixels, mm)
Desktop	69	44	16	9
Handheld	21	3	9	9
Automotive	35	11	14	10
Head-mounted	57	37	2	18



(**a**) (**b**) (**c**) (**d**)

Figure 1. Eye gaze applications in various consumer platforms (from left: (**a**) desktop [2], (**b**) tablet [3], (**c**) automotive [4], (**d**) head-mounted [5] eye tracking setups).

Another sparsely investigated issue in gaze research is about studying the impacts of different error sources (i.e., system and user variables) on gaze tracking performance. It was highlighted in [1] that gaze estimation systems on various user platforms face varied operating conditions, which are not well described in conventional research or product literature. Such factors which possibly influence a gaze estimation system's accuracy include user distance, display properties of the screen where gaze is tracked and user head pose variations. Additionally, for eye tracking on mobile platforms like handheld devices, platform motion and orientation may greatly alter the claimed accuracy of an eye tracker during real life operations. At present, research works commonly do not report impact of these factors or present their results in any uniform graphical format.

1.1. Problem Statement

As described above, there are definite requirements of methods, firstly for detailed analysis of eye tracking data to learn about the tracker system characteristics under variable operating conditions and secondly for attaining homogeneous and fully defined gaze accuracy metrics. Therefore in this work, for comprehensive evaluation of generic eye trackers, a set of metrics and visualization methods are derived by analyzing the data collected from an eye tracker. To develop these metrics and visual tools, experiments with the eye tracker are performed with a group of subjects on two different user platforms (desktop and tablet). The gaze experiments are done on the desktop and tablet devices under controlled variation of tracking conditions such as changes in user distance, user head-pose and platform orientations to study their impacts on gaze tracking accuracy of the tracker. Using the gaze data collected from experiments, several metrics are derived which not only specify tracking accuracy but also look for specific trends in the gaze data that indicate ways to achieve best tracking performance from the tracker. Visualization tools are then built by aggregating the experimental data to understand and compare tracking performance under these varying circumstances. Finally, all the described metrics and visual methods are built into a graphical user interface software with which users can readily explore their own collected gaze datasets, study their tracker characteristics and save the results without delving into details of the source code.

1.2. Purpose, Scope and Structure of the Paper

The purpose of this paper is to introduce some metrics and visualizations that may be helpful to determine the data quality from generic eye trackers when they operate under variable operating conditions. For this, a series of experiments are done to collect data from a commercial eye tracker and

its operating conditions were varied, to build specific metrics and visuals that reflect these conditions. Two platforms were used, a desktop and a tablet mounted with the eye tracker to collect gaze data using the same stimuli and data logging process. The tablet was used to especially study the impact of platform orientation variations which is unique to this platform. The metrics defined here were applied on the collected data and the results are used to demonstrate how these metrics may be used by any eye tracker. Similarly, the visualizations are meant to show how these may be used in any gaze tracking experiment to visually inspect gaze data quality and aggregate test results.

At this point it is also important to outline the scope of this work. It may be noted that in this work we do not intend to evaluate a particular eye tracker. We neither want to classify the used tracker or its data as good or bad or determine their suitability for any application. In other words, here we are not judging an eye tracker itself, but are only using eye tracking data to implement our proposed metrics and visuals. We note that eye tracker data patterns change with varying operating conditions and we expect the developed metrics to represent these changes quantitatively and the visualizations to show this graphically. Aspects like why the tracker behaved in a certain way under certain operating conditions, or analyzing if the accuracy levels are acceptable, are out of scope for this work. This is because analyzing the underlying reasons for data variability is a dedicated body of research, but comprises of a different direction of work with respect to the topic presented here.

As for the structure and organization of the paper contents, there are two major sections here, Section 4 on the Metrics and Section 5 on the Visualizations. Section 4 has several Sections 4.1–4.4 and sub-subsections such as Sections 4.1.1, 4.1.2, 4.2.1, 4.2.2 and 4.3.1, Sections 4.3.2–4.3.4. Each of these sub-subsections present a gaze data evaluation metric and corresponding results from implementing the metric on gaze data collected from relevant experiments. For each metric description, the content is split into "Method" which describes the concept of the metric and "Example results" which displays the outcome from testing the metric on collected data in tabular or graphical form. Similarly, in Section 5 the contents of Sections 5.1.1–5.1.5 and 5.2.1, Sections 5.2.2–5.2.4 are split into "Method" and "Example results". A summary of related research is provided in Section 2, the experimental details in Section 3 and conclusion to this work is in Section 6.

2. Related Works: Accuracy Estimates and Visualizations in Gaze Research

In this section, the status of accuracy metrics and data visualizations used in contemporary eye gaze research works is reviewed in Sections 2.1 and 2.2, and the contributions of this work in this direction are highlighted in Section 2.3.

2.1. Previous Work on Evaluation of Gaze Estimation Systems

Eye gaze research has progressed to include multiple user platforms with increasing number of applications in human computer interactions [6–8] marketing, psychology and ecommerce [9,10]. A major aim of eye tracking system design is to achieve high accuracy and consistency in results. A wide range of eye tracking systems and algorithms have been developed over the past few decades which have been reviewed in [11,12]. Dedicated research works on evaluation and reporting of gaze data quality and standardization of has been presented in [13–16]. More works such as [17–19] discuss on the systematic performance evaluation of eye tracking systems including frameworks, databases and virtual environments.

Reference [13] discusses the lack of common standard measures in for eye data quality which affect eye tracker usage and research on eye movements. It provides a detailed description of what is meant by eye data quality and how data accuracy and precision affects measurements of eye movement features such as dwell time, number and duration of fixations and pupil size and further describes standardized means of reporting data quality. Reference [14] describes the use of an artificial eye to implement methods for objective evaluation of eye tracking data quality. First, the temporal accuracy and latency of an eye tracker is tested using an artificial saccade generator. Then an artificial pupil is mounted on a computer controlled moving platform to provide biologically similar eye movements.

Reference [15] studies the impact of eye data collection conditions on the factors such as calibration, pupil identification, fixation detection and gaze analysis. They implement mobile eye tracking in an outdoor environment and report that all stages of processing eye tracking data must be tailored to the data collection conditions. Reference [16] discusses the need for development of a standard application programming interface API for eye trackers in order to build applications using gaze data. The aim is to have a unified way to interact with eye-tracking systems and receive gaze data using same protocol regardless of the eye tracker.

Works such as [20–24] present results from the evaluation of specific commercial eye trackers from companies such as Eyetribe, SMI, Tobii and GazePoint. Reference [20] compares the accuracy, precision and sampling rates of the Eye Tribe (The Eye Tribe ApS, Copenhagen, Denmark) and SMI RED eye trackers (SensoMotoric Instruments, Teltow, Germany) and shows the impact of system setups such as user sitting position, height of stimulus screen, height of eye tracker, user tracker distance and frame rate on the data from the trackers. Reference [21] also discusses the evaluation and comparison of the EyeTribe and SMI RED eye trackers by concurrently recording data from them and measuring parameters like accuracy, data loss and fixation counts for application in cartographic research. Reference [22] presents the comparison of several eye trackers such as Eye Tribe, Tobii EyeX (Tobii Technology Inc., Danderyd, Sweden), Seeing Machines faceLAB (Seeing Machines, Fyshwick, Australia), Smart Eye Pro and Smart Eye Aurora (Smart Eye AB, Gothenburg, Sweden) to study features such as gaze tracking accuracy, precision, impact of glasses and data loss. Reference [23] discusses the objective evaluation of the Gazepoint GP3 eye tracker (Gazepoint Research Inc., Vancouver, BC, Canada), and studies its capabilities with respect to pupil dilation metrics under cognitive loads and luminance conditions, whereas [24] evaluates the Tobii EyeX tracker for accuracy, precision and latency parameters to determine its suitability for behavioral studies.

However, from these works, it may be observed that apart from accuracy, precision and latency, no other evaluation metrics for eye tracker evaluation have been studied in detail. Also within these works there remains inhomogeneity in the definition of performance metrics as some of them report gaze accuracy in degrees while some in pixel measures. These aspects have been discussed in our previous paper [1] which describes the existing inhomogeneity in gaze data accuracy measures and the need for development of more intricate gaze tracking performance metrics. It also proposed the concept of a dedicated evaluation framework for all round performance assessment of eye trackers.

2.2. Data Visualizations in Gaze Research

In [25] a comprehensive overview and classification of gaze data visualization methods is presented. The methods are grouped on the basis of visualization type (e.g., statistical/spatio-temporal, 2D vs. 3D visualizations), eye tracking data type (fixation, scanpath, smooth pursuit, saccades) and stimuli (point-based for studying gaze distribution and areas of interest (AOI)-based for understanding AOI interrelationships). Other classifications include: static (image based) and dynamic (video based), active and passive stimulus based visualizations. Conventionally gaze data is aggregated using heat maps and fixation maps, on which several improvements have been proposed, such as in [26] (modifying transparency of the heat map depending on gaze data) and [27] (real-time heatmap generation and visualization of 3D gaze data). In [28], gaze visualization with parallelized rendering on a GPU (graphics processing unit) platform is proposed. The aim is to speedup heatmap creation on video stimulus, while [29] proposes dynamic heatmap visualizations coupled with user visual focus information on different backgrounds (dark, blurred, fog). Several novel visual approaches have in been reported in [30–32]. In [30,31] gaze stripes and color bands are described in which a sequence of gaze point images are aligned in a timeline to analyze and compare the viewing behavior of multiple participants in a clutter-free manner. Kurzhals et al in [32] presents a visual analytics method to interpret eye movement data recorded for dynamic stimuli such as video or animation. Scanpath is an useful gaze data visualization method which is used in [33] to differentiate scanning behavior of participants over stimulus images. It is a very detailed work on studying viewing patterns of users

looking at natural scene images by applying scanpath comparison metrics like string edit distance, sample based, linear distance etc. on eye tracking data. Other innovative gaze data representations can be found in [34,35] and a popular open source gaze data analyzer is described in [36].

The above literature survey shows that a wide range of state-of-the-art methods for gaze data analysis and techniques for visualization are currently present in gaze research. However, it can be seen that these methods are mostly directed towards exploration of eye movement characteristics (such as speed, direction and duration), understanding its relation to human behavior [37] such as attention, cognitive load assessment, regions and sequence of interests [38] or studying visual saliency [39]. No visualization work is found which is dedicated towards evaluation of eye trackers themselves, or studying the data characteristics of eye trackers under variable operating conditions.

2.3. Requirement of Well-Defined Accuracy Metrics and Performance Visualization Tools in Eye Gaze Research: Contributions in This Work

It is observed that currently there is a lack of well-defined metrics for comprehensive evaluation of eye trackers and standard open source visualization methods for understanding their performance characteristics are also not available. Keeping these in mind, this paper puts forward some effective solutions for current and future gaze researchers for complete characterization of their eye tracking devices. The contribution in this paper includes the development and description of a set of accuracy metrics and visualization methods, and a software user interface for accessing these to produce and save results, with all of these being meant for use on data from any generic eye/gaze tracker to completely specify its performance.

Benefits of the metrics presented in this paper include: (a) derivable simply from eye gaze spatial coordinates and corresponding ground truth data (b) provide quantitative measures of an eye tracker's performance and therefore can be used to compare multiple eye tracking systems and algorithms (c) can help to estimate impact of operating conditions on eye tracking data (d) can be adapted to different computing systems, display sizes and resolutions (e) may be used irrespective of eye tracking algorithm and hardware (f) reveal specific trends in gaze data which indicate ways to improve tracking performance.

With the proposed visualization methods, one may: (a) have a quick look at an eye tracker's data characteristics without going deeper into the tracking system/algorithm (b) visually compare tracking performance and data quality of multiple eye trackers (c) present volumes of experimental data in single or few figures. With the graphical user interface named GazeVisual developed in this work, which is described in more detail in later sections of the paper, a generic user of any gaze based application may implement all the metrics and visualization functions described in this work on their gaze data without going into details of source code. Also, owing to the open source nature of the software, it may be adapted by eye gaze researchers to suit their individual research purposes, while advanced programmers may also contribute towards its functional extensions.

3. Experimental Methodology

3.1. Experimental Setup and Workflow

Eye tracking experiments are performed on desktop and tablet platforms for gaze data collection. The setup details are provided in Table 2 and comprises of a remote eye-tracker mounted on the screen of a desktop or a tablet device, which display a visual stimulus interface (UI). In the UI, a moving dot sequentially traces a grid of (5×3) locations over the display screen as shown in Figure 2a (it shows the static view of the screen locations traced by the dot). The UI dot radius is 10 pixels and it stops at each location for 3 s before moving on to the next. The angular extent of the UI stimulus grid is 30 degrees of visual angle at 45 cm distance. The UI is synchronized with the eye tracker to collect gaze data during an experimental session. The locations traced by the dot are henceforth called areas of interest or AOIs whose on-screen positions are known in pixel coordinates. The collected data comprises of

a participant's gaze coordinates on the display and corresponding time stamps as estimated by the tracker, while AOI locations form the ground truth. All data are stored in comma separated values (CSV) files.

Experiments are done by positioning users in front of a computer screen mounted with the tracker while their head is fixed with a chin rest. During the experiments, the UI is run on the desktop or tablet screen and users are asked to follow the moving dot as it moves. This ensures that the user's fixation distance is closest to stimuli locations. The eye tracker calibration uses 9 points and the calibration stimulus comprises of dots appearing at the corners, top and bottom locations of the display. The AOI stimulus dot size is comparable to the eye tracker calibration stimulus dot. After the calibration procedure, the calibration quality is validated using a validation procedure provided by the eye tracker software, and for poor calibration, the process is repeated.

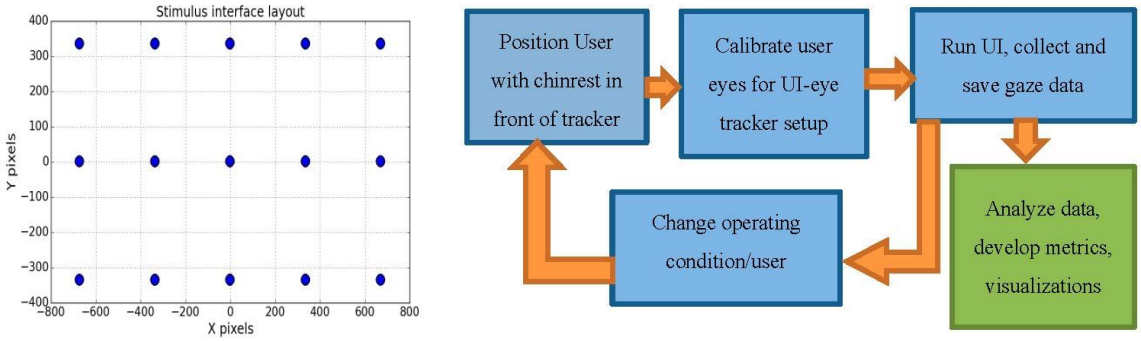


Figure 2. (**a**) Shows the layout of the stimulus interface (UI) with AOIs (circles) where a user has to look during data collection. (**b**) Shows the experiment flowchart for data collection.

A typical experimental workflow is shown in Figure 2b. The procedure followed during one complete experimental session is presented in the top three blocks. After completion of each session, certain system or user parameter is changed, such as registering a new user, varying user distance or head pose, and then the experimental sessions are repeated with the new condition. Outcome of the experiments is the collected gaze dataset (block on bottom right corner) which is then analyzed and used for development of metrics and visualizations for performance evaluation of the eye tracker.

Table 2. Experimental setup details.

Eye Tracker and UI setup	Details	Display and Hardware Characteristics	Details	Experimental Variables
Tracker type	Desktop based, NIR LEDs + 1 Camera, 3 ps	**Screen Size**	Desktop: 22 inch Tablet: 10.1 inch	Single user, multiple user
Calibration	6 point	**Screen Resolution**	Desktop: 1680 × 1050 Tablet: 1920 × 1200	Fixed and variable user distance
Tolerance	Maximum user distance: 80 cm, spectacles allowed, chin-rest used	**Screen properties**	Desktop: 21.5 inch diagonal, width × height = 18.5″ × 11.5″ Tablet: 10.1 inch diagonal, width × height = 8.5″ × 5.5″	Fixed and variable head pose
User interface	15 AOI locations, AOI radius: 10 pixels	**Pixel sizes of desktop and tablet screens**	Desktop: 0.275 mm Tablet: 0.113 mm	Screen resolution and pixel size
Eye data type	Fixation, AOI duration: 3 s, blinks allowed between AOIs	**Hardware details for desktop and tablet**	Desktop: Core i7, 3.6 GHz, 16 GB ram Tablet: Intel Atom X5, 1.44 GHz,4 GB ram	Platform orientation

3.2. Setup Coordinate System and Eye Movements

Figure 3a,b show the experimental environment, while Figure 3c explains the setup coordinate system in which gaze data is collected during experiments. The display screen is the area used to show the stimulus points from both the UI and tracker calibration routines. The display coordinate system is aligned with the display of the desktop or tablet used in this work and its origin is the upper left corner of the display screen. The eye tracker coordinate system has its origin at the center of the frontal surface of the eye tracker which is aligned with the center of the display screen. The tracker x-axis points horizontally towards the user's right, the y-axis points vertically towards the user's up and the z-axis points towards the user, perpendicular to the front surface of the eye tracker. The gaze data comprise of eye locations of a user tracked by the eye tracker and mapped into the 2D coordinates of the display screen. The gaze x, y data of user eye locations using this coordinate system has (0,0) at the display screen center and z data represents the user distance from the tracker starting from 0 at the tracker.

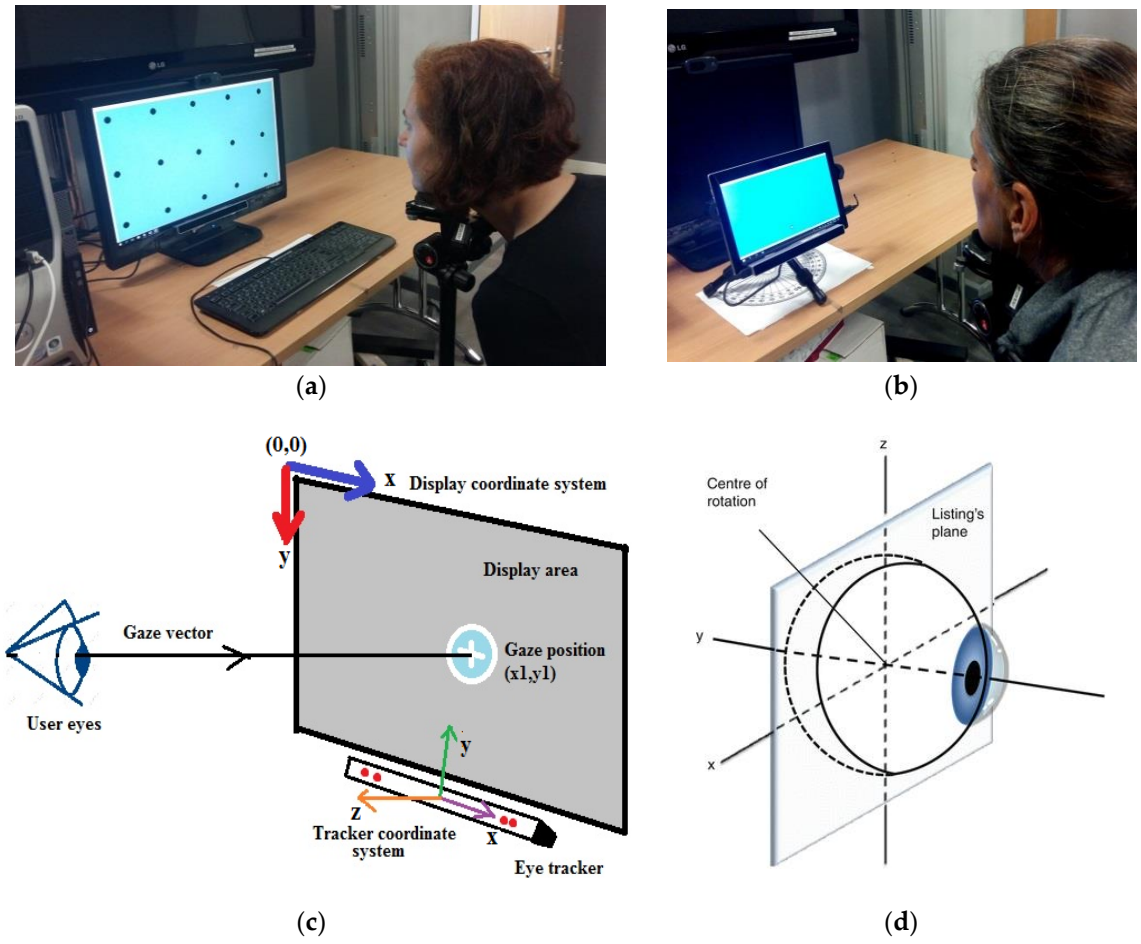


Figure 3. The figure on top left (**a**) shows the eye tracker under test mounted on a desktop computer display along with a participant seated in front of it (desktop screen shows the AOI layout in static form). (**b**) Shows a tablet mounted with the eye tracker using the same UI and experimental workflow as that used for the desktop. (**c**) Shows the display and eye tracker coordinate systems for this work and (**d**) shows the eye movement coordinate systems with Listing's plane and Fick's axes.

Describing the kinematics of the eye requires the definition of reference positions and coordinate systems. Primary position of the eye is one such reference and defined as the position the eye assumes when the subject is looking straight ahead, while the head is kept upright. Movements of the eye around the primary position may be defined using several coordinate systems such as Fick's, Helmholtz

and Euler [40]. Out of these, in this work, the Fick's coordinate system is assumed for describing eye movements along with the Listing's plane (Figure 3d). The axes of Fick have a head-fixed vertical axis and eye-fixed horizontal axis [41]. In the Fick's axes, the x-axis is the transverse axis passing through the center of the eye at the equator and vertical rotations of the eye occur about it. The y-axis passes through the pupil and torsional rotations occur about this axis. The z-axis is a vertical axis; and horizontal rotations occur about this. Listing's equatorial plane contains the center of eye rotation and the x and z axes while the y-axis is perpendicular to it. Eye-fixed reference frames as the Fick's axes [42] are similar to mechanical mounting system like gimbals where one axis is for panning left or right (yaw or horizontal axis) and one for tilting up or down (pitch or vertical axis). Torsional movements of the eye are not considered in this work.

3.3. Eye Tracking Experiments Conducted for the Development of Gaze Accuracy Metrics and Visual Tools

The purpose of the experiments described in this work is to collect eye tracking data under variable operating conditions that may affect an eye tracker working on a desktop or tablet platform. The collected data is used to implement and test evaluation metrics and visualizations and described in Sections 4 and 5. The following experiments were conducted in this work. (a) User distance variability experiments: In these experiments, gaze data is collected with user-eye tracker distances of 45, 60 and 75 cm. The terminologies "UD45", "UD60", "UD75" are used in this paper for referring to experiments/datasets obtained from the tracker at the distances of 45, 60 and 75 cm, respectively. (b) Head-pose variability experiments: This is relevant to studying the effect of a user's head pose on gaze tracking accuracy of the eye tracker. By head pose, the position of a user's head in 3D space in terms of roll, pitch and yaw (RPY) angles is meant here. During the experiments, a user is seated at a fixed distance (60 cm) from the tracker and is asked to vary their head position to different rotation angles (head pose in roll, pitch, yaw) while looking at the UI on the display screen and their gaze is tracked on the UI. The head position is also tracked simultaneously using a head pose model that measures head pose angles in RPY values with 1 degree of accuracy. Gaze tracking errors corresponding to different head-poses are then analyzed (c) Platform orientation experiments: Eye tracking on dynamic platforms like tablets face some unique challenges since their positions vary frequently and result in variable orientation of eye trackers which are mounted with the tablet screen. To quantize the impact of tracker orientation on gaze data, experiments are performed in which the orientation of the tablet device mounted with the eye tracker is varied with respect to the user at fixed platform roll, pitch and yaw angles. Eye tracking data is collected for each tablet orientation with the same test UI as used for the desktop system. The objective of these experiments is to study impact of platform orientation variations on eye tracking data characteristics.

4. Deriving Evaluation Metrics for Eye Tracking Devices and Algorithms

Eye tracking accuracy is typically measured in terms of the difference between the real stimuli positions and the corresponding measured gaze positions and expressed as angular, pixel or distance measures. However, accuracy expressed in this way provides little information about detailed tracker characteristics and impact of variable operating conditions. Therefore, a set of metrics are derived and presented here, which aim at describing the quality of eye gaze data by taking the characteristics of a gaze tracking system into consideration. The metrics derived in this work are classified into four categories, namely angular accuracy metrics, statistical metrics, sensitivity metrics and a new metric based on Receiver Operating Characteristic (ROC), which are described in the subsections below.

4.1. Angular Accuracy Metrics

4.1.1. Gaze Angular Accuracy

Different research groups working in eye gaze most often use independent accuracy metrics or do not describe their accuracy calculation in detail. In order to facilitate the interpretation of a generic

eye tracker's specifications, as well as to provide an objective way to compare different systems, it is important that the same accuracy metrics are used by everyone and each metric be clearly described. That is, eye tracking data must be analyzed in a standard and consistent way. The purpose of this section and its sub-sections is to describe such a common set of calculations, which may be used to measure and compare the accuracy of eye trackers by using only their raw data outputs (and ground truth locations), irrespective of their tracking algorithm or platform.

Method:

Starting from raw gaze x,y pixel coordinates of the left and right eye (X_{left}, Y_{left} & X_{right}, Y_{right} respectively) obtained from the tracker, the angular accuracy of gaze tracking is derived below [43]:

Gaze Point Coordinates in Pixels:

$$\text{GazeX} = \text{mean}\left(\frac{X_{\text{left}} + X_{\text{right}}}{2}\right), \text{GazeY} = \text{mean}\left(\frac{Y_{\text{left}} + Y_{\text{right}}}{2}\right) \quad (1)$$

Gaze Position in mm of on Screen Distance:

$$\text{XPos (mm)} = \mu * \text{GazeX}, \text{YPos (mm)} = \mu * \text{GazeY} \quad (2)$$

where μ is the pixel size of the particular monitor which is calculated depending on the monitor screen dimensions and pixel resolution. The calculation for the factor μ is shown as: $\mu = d_m/d_p$ where d_p is the screen diagonal size in pixels as obtained from Equation (3) below, w_p is the screen width in pixels, h_p is the screen height in pixels and d_m is the diagonal size in mm (converted from inches):

$$d_p = \sqrt{w_p^2 + h_p^2} \quad (3)$$

For example, when our experiments were performed on a 22 inch (diagonal) monitor operating at 1680×1080 pixel resolution, we had $d_p = 1981$, $d_m = 558.8$ and $\mu = 0.28$.

On Screen Distance (OSD):

When the origin of the gaze coordinate system is at (x_{pixels}, y_{pixels}), the on-screen distance of a user's gaze point is the distance between the origin and a certain gaze point. It is given by Equation (4), with the offset being defined as the distance between the tracker sensor and lower edge of display screen. In our case, the tracker is attached directly below the screens so the offset value is 0 and x_{pixels},y_{pixels} = (0,0), i.e., origin is the center of the screen:

$$\text{OSD (mm)} = \mu\sqrt{\left(\left(\text{GazeX} - \frac{x_{\text{pixels}}}{2}\right)^2 + \left(y_{\text{pixels}} - \text{GazeY} + \frac{\text{offset}}{\text{pixelsize}}\right)^2\right)} \quad (4)$$

Gaze Angle Relative to the Eyes:

Using trigonometry, the gaze angle of a point on screen relative to a user's eyes is calculated as:

$$\text{Gaze angle } (\theta) = \tan^{-1}(\text{OSD}/Z) \quad (5)$$

where Z is the distance of the eye from the screen. The distance between the eye and the gaze-point (mentioned as EGP) is estimated from 3D Cartesian geometry as:

$$\text{EGP (mm)} = \sqrt{((\text{GazeX})^2 + (\text{GazeY})^2 + (Z)^2)} \quad (6)$$

Pixel Distance between Ground Truth and Estimated Gaze Point (pix_dist):

As described in Section 3, the AOI locations (x, y coordinates) displayed on the screen during the experiment form the ground truth for data collections. Using Cartesian geometry, the shift between the ground truth coordinates (GT.X, GT.Y) and tracked gaze locations (GazeX, GazeY) is given by:

$$\text{pix\_dist (pixels)} = \sqrt{((\text{GT.X} - \text{GazeX})^2 + (\text{GT.Y} - \text{GazeY})^2)} \quad (7)$$

Angular Accuracy:

The gaze estimation accuracy (or error as referred in this paper) of an eye tracker is expressed in absolute units (degrees) as the angular deviation between ground truth and estimated gaze locations. Using the estimates of gaze angle, pixel distance and the distance between eye and the gaze point from Equations (5)–(7), the formula for estimating gaze tracking accuracy may be calculated as:

$$(\mu * \text{pix\_dist} * \cos(\text{mean}(\theta))^2)/\text{EGP} \quad (8)$$

Example Results:

Using these equations and data from our experiments, the mean angular accuracy of the eye tracker used in this work is found to be between 3 to 5 degrees for a user tracker distance of 45 cm, 2 degrees for a distance of 60 cm and 0.9 to 2 degrees for 75 cm respectively. The results from the above calculations are also used to estimate gaze error throughout this paper to compute other gaze data metrics and implement visualizations.

4.1.2. Gaze Yaw and Pitch Angular Accuracies

The calculations in Section 4.1.1 indicate user gaze angular accuracy by considering a primary eye position. The eye also undergoes rotational motion which leads to different eye orientations relative to the head. The two eye rotation variables are gaze yaw and pitch, where the yaw variation corresponds to left-right and pitch variation corresponds to top-bottom eye movements.

Method:

The gaze yaw and pitch angles are derived as follows:

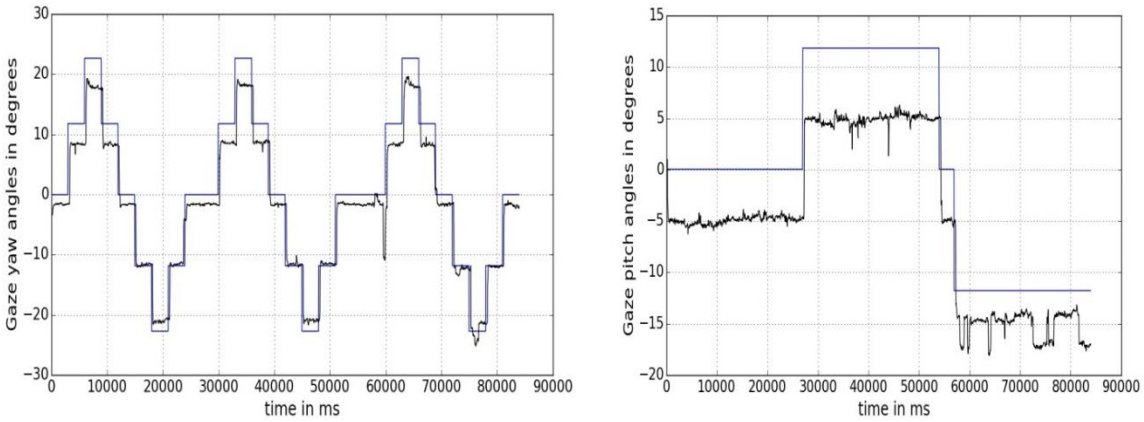
$$\text{Gaze pitch}(\theta\text{pitch}) = \tan^{-1}(\text{GazeY}/\text{Z}),\ \text{Gaze yaw}\ (\theta\text{yaw}) = \tan^{-1}(\text{GazeX}/\text{Z}) \quad (9)$$

where, GazeX, GazeY and Z are defined above in Equations (1), (2) and (5). To estimate the gaze yaw and pitch errors from the experiments, the ground truth yaw and pitch angles are first calculated using the position of the AOI dots as they appear and move on the screen. The ground truth pitch and yaw value for each AOI dot with screen coordinates (AOIx, AOIy) are given by:

$$\text{AOI pitch} = \tan^{-1}(\text{AOIy}/\text{Z}),\ \text{AOI yaw} = \tan^{-1}(\text{AOIx}/\text{Z}) \quad (10)$$

Example Results:

Using Equations (9) and (10), the gaze yaw and pitch angle values along with ground truth yaw and pitch values for one user gaze data during one experimental session are plotted against time in Figure 4a,b, respectively. In Figure 4a, the blue line represents the ground truth yaw angles, as calculated from Equation (10). Each step in the curve represents a different AOI position on the screen. The black line shows the variation of a user's gaze yaw angle with time as estimated by Equation (9). In Figure 4b, ground truth AOI pitch angles are shown in blue and estimated gaze pitch angles in black.



(**a**) (**b**)

Figure 4. (**a**) On the left shows the gaze yaw angle variations (overlaid with ground truth) with time as recorded for one person during one complete experimental session (i.e., time starts when the user starts gazing at AOI locations appearing on the screen and stops at the last AOI). (**b**) On the right shows the gaze pitch (and ground truth) variations during one session.

The maximum and minimum (±) gaze yaw and pitch values for 5 users seated at 45 cm from our tracker were found to be ±22 degrees and ±12 degrees respectively. Gaze pitch errors are seen to be higher in magnitude than gaze yaw error values, and are about 4.5 degrees at 45 cm while yaw error is about 2.6 degrees. Thus, gaze tracking errors are not just scalar values but also have directional components which are not reflected if only mean error values are considered.

4.2. Statistical Metrics

4.2.1. Statistical Measures of Eye Tracking Performance

Gaze experiments are usually performed on a group of subjects with group sizes ranging between 5–15 or even 20–30 participants for improving test reliability. For analyzing gaze data from the numerous subjects, relevant statistical parameters on the collected data must be evaluated to draw significant inferences on collective data characteristics and insight into error patterns.

Method:

The following statistical parameters were used to evaluate our eye tracking system:

$$\text{Mean } (\varphi):\ 1/n\sum_{i=1}^{n} x_i,\ \text{Z score } (\sigma):\ (x_i-\varphi)/\sigma, 95\%\ \text{confidence}:\ \varphi\ \pm 1.96\sigma/\sqrt{n}$$

where σ is the standard deviation and n is the number of data points. The 95% confidence interval signifies the certainty that the mean error value would lie within the upper and lower bound of this interval. A larger confidence interval indicates higher variability in the data, whereas a small interval indicates more consistency of results. The Z score indicates the presence of unusual data points within the dataset, as gaze data quality may vary from person to person. It might be noted that typical research works in eye gaze rarely analyze their results using detailed statistical methods other than specifying the mean error values.

Example Results:

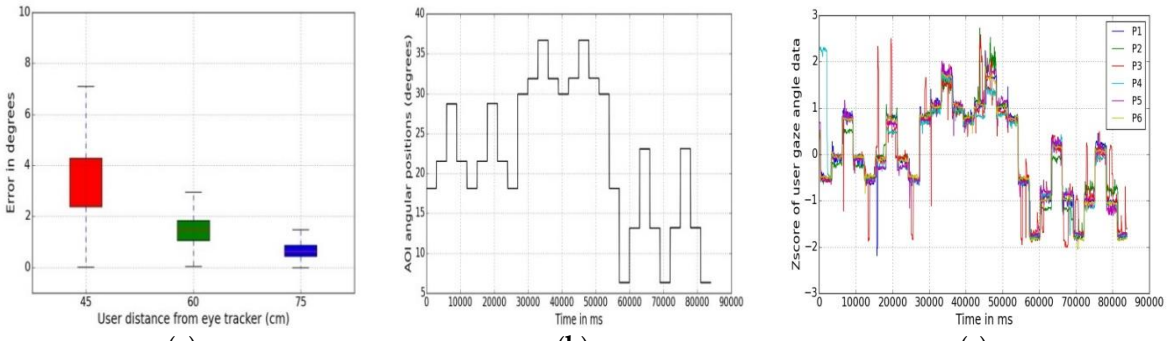
Examples of statistical analysis on gaze data are shown in Table 3 and Figure 5a–c. Data collected from user distances experiments (for 45 and 75 cm) are used for analysis. Implementing the metrics on gaze data reveals some important characteristics of the eye tracker under the impact of variable user distance conditions. It is seen that maximum gaze angle and mean gaze errors are higher when

users are closer to the tracker (at 45 cm) than when they are further away (75 cm). Also, the confidence intervals are lower in UD75 than in UD45 experimental data, which means that error variability is more at lower user-tracker distances. Use of statistical metrics may therefore help to know how to get optimal good accuracy and better consistency of results from any given tracker.

Table 3. Gaze data error statistics for two user distances, data from same five users.

User Distance = 45 cm	Max Gaze Angle (degree)	Mean Error (degree)	95% Confidence Interval
User 1	34.24	3.63	3.54–3.71
User 2	32.83	3.96	3.88–4.04
User 3	39.04	3.42	3.35–3.49
User 4	32.99	4.61	4.51–4.70
User 5	34.69	3.52	3.44–3.59
User Distance = 75 cm	**Max Gaze Angle (degree)**	**Mean Error (degree)**	**95% Confidence Interval**
User 1	22.97	0.91	0.85–0.96
User 2	23.36	0.98	0.93–1.02
User 3	24.28	0.94	0.89–0.99
User 4	22.47	1.84	1.79–1.89
User 5	23.36	2.08	2.02–2.12

A box plot is a valuable method to compare several error statistical attributes from multiple experimental datasets [44]. A box plot of the data from the three user distance experiments (45, 60, 75 cm) is shown in Figure 5a. It presents the statistical result summary, i.e., minimum, first quartile, median, third quartile, and maximum error values for the UD45, UD60 and UD75 experiments in a single figure. In each box the minimum and maximum data values represent the endpoints of the vertical line and height of the box shows interquartile range. A small interquartile range indicates that data is centered while a larger range means that the data is more scattered. The result of using box plot on UD45 and UD75 data indicates that UD45 dataset is more scattered in nature than UD75 data, and the error magnitudes are also higher. Another statistical metric, called Z scores analysis is done on UD45 gaze data to estimate the level of scatter in data points at different AOI positions and is shown in Figure 5c, which helps to reveal person-to-person variation of gaze data properties.



(**a**) (**b**) (**c**)

Figure 5. These plots demonstrate the use of statistical metrics to evaluate data from eye tracking experiment. (**a**) Shows a box plot of gaze data from three experiments UD45, UD60, UD75. (**b**) Shows the variation of ground truth angular positions of the AOIs during one experimental session done at 45 cm, as a function of time. (**c**) Shows the corresponding Z score variations of gaze angles for 5 users (P1–P5). The Z score represents how many standard deviations each data point is away from the mean and therefore shows the number of unusual points or outliers in the dataset. (**c**) Shows that level of gaze data scatter varies from person to person for same experimental conditions.

4.2.2. Histogram Based Metrics

To study the similarity between data from different eye trackers, experiments or comparison of data from a test experiment with a reference dataset, certain data similarity metrics based on histograms can be used. Such metrics for histogram comparison include correlation [45], intersection [46] and Bhattacharya distance [47].

Method:

If H_1 and H_2 are the histograms gaze errors from two datasets from the experiments, then d(H_1, H_2) is the histogram comparison metric and N is bin size. For the correlation and intersection measure, higher the result, the more accurate is the match whereas for Bhattacharya distances, lower the result, the better the match. The expressions for three different similarity measures for comparing two histograms are given below:

$$\text{Correlation}: \ d(H_1, H_2) = \frac{\sum_I (H_1(I) - \overline{H_1})(H_2(I) - \overline{H_2})}{\sqrt{\sum_I (H_1(I) - \overline{H_1}) \sum_I (H_2(I) - \overline{H_2})}} \text{where } \overline{H_k} = \frac{1}{N} \sum_J H_K(J) \quad (11)$$

$$\text{Intersection}: \ d(H_1, H_2) = \sum_I \min(H_1(I), H_2(I)) \quad (12)$$

$$\text{Bhattacharya distance}: \ d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{H_1 H_2 N^2}} \sum_I (H_1(I) - H_2(I))} \quad (13)$$

Example Results:

Using these three metrics, the similarity measures between the gaze error data from a pair-wise choice of the user distance experiments (e.g., UD45-UD60, UD45-UD75, total 6 pairs) are calculated and presented below in Table 4. The values in each cell are the results obtained from using the above Equations (11)–(13) on a pair of histograms calculated on gaze error data from the experiments. It is observed that match between data from UD45 and UD75 experiments is low. Results from UD60 and UD75 experiments have better correspondence. This metric could be especially useful to compare data from multiple eye trackers captured under a wide range of operating conditions, in which it is difficult to assess data characteristics or correspondence by looking at just the error magnitudes.

Table 4. Histogram comparison results between different user distance datasets.

Similarity Metric	Datasets Taken at User Distances 45, 60, 75 cm					
	45–45	**45–60**	**45–75**	**60–60**	**60–75**	**75–75**
Correlation	1.0	0.31327	0.18693	1.0	0.61210	1.0
Intersection	2457.07	1131.38	970.69	2502.46	1565.38	2514.01
Bhattacharyya	0.0	0.47526	0.52651	8.10×10^{-10}	0.34091	8.10×10^{-10}

4.3. Sensitivity Metrics

Gaze estimation systems operating in real life face several non-ideal conditions like changes in user head pose, user distance, variation in display size and resolution and if eye tracking is done on a mobile platform, then variation in platform orientation as well. Currently, there exist no quantitative metric to indicate how the accuracy of an eye tracker may be affected due to presence of each of these factors. Therefore, by running experiments on these factors (head pose, user distance, platform orientation variations, display characteristics) and using the collected data, several gaze-error-sensitivity metrics are derived, tested and presented in the subsections below.

4.3.1. Head Pose Sensitivity

Head pose sensitivity analysis is done in this work to determine how the variations of a user's head pose may affect the gaze tracking accuracy of a tracker under test. This is because the manufacturers or designers building the eye trackers do not quantitatively specify how much the tracking accuracy may deteriorate if user head pose is varied. For deriving this metric, data is used from our head-pose variability experiments as described in Section 3 above.

Method:

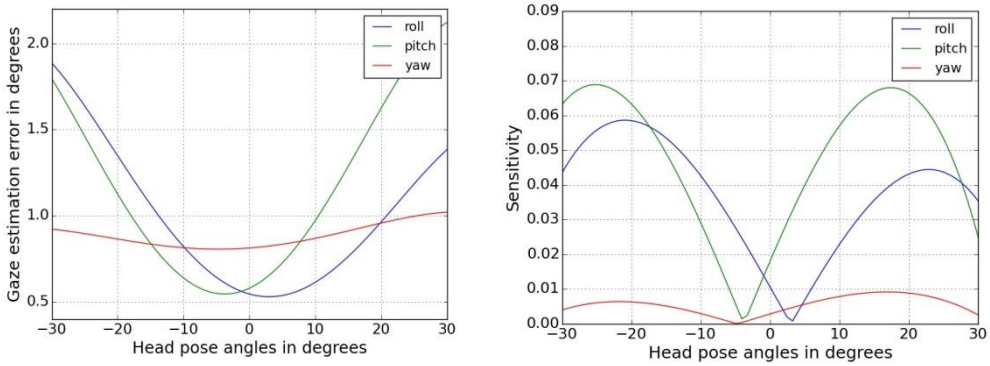
The head pose sensitivity for head pose variation in roll, pitch, yaw (r, p, y) angles is defined as:

$$S_r = \frac{\partial E_r}{\partial H_r},\ S_p = \frac{\partial E_p}{\partial H_p},\ S_y = \frac{\partial E_y}{\partial H_y} \tag{14}$$

where S_r, S_p and S_y are the sensitivities with respect to head pose roll, pitch and yaw angles respectively. H_r, H_p and H_y are the head pose angles in roll, pitch and yaw directions and E_r, E_p and E_y are the corresponding gaze estimation errors (in degrees). The plot of gaze error vs. head pose angles is shown in Figure 6a and the head-pose sensitivity plots for head pose changes in roll, pitch and yaw directions are shown in Figure 6b.

Example Results:

From the head pose sensitivity experiments, it is seen that lowest errors occur for frontal head positions and sensitivity increases as magnitude of the head pose angles increases. Figure 6a also shows that the roll and pitch component of head motion affects gaze errors more strongly than yaw movements. Overall this metric helps to show that the head pose tolerance of the given eye tracker is quite low, and to achieve reliable gaze tracking, a user head movements must be constrained, and therefore the use of chin rest is essential.



(**a**) (**b**)

Figure 6. (**a**) Shows the variation of gaze estimation error as a function of head pose angles in roll, pitch, yaw directions. (**b**) Shows the variation of head pose sensitivity as a function of head pose. These plots help to know how much head movement is allowed to keep gaze error within limits.

4.3.2. Platform Orientation Sensitivity

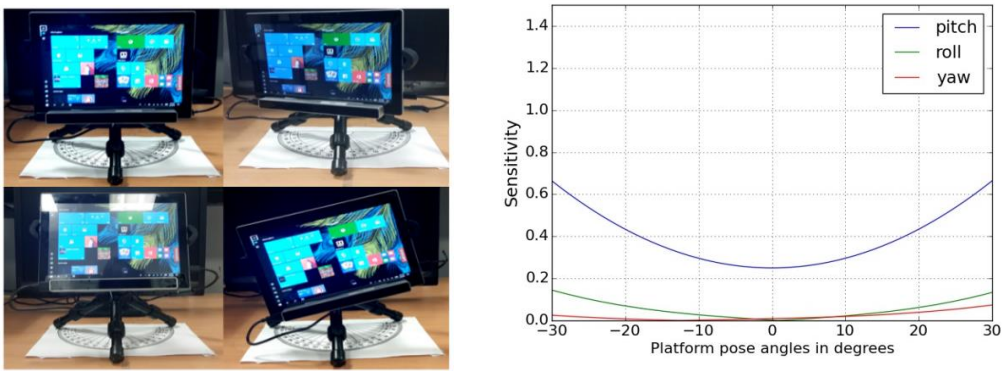
This is a new kind of study in which the impact of platform orientation on the accuracy of an eye tracker mounted on a tablet platform is observed. This is a vital analysis for eye tracking and gaze applications on handheld devices like smartphones and tablets, which face reliability issues due to the highly dynamic nature of these devices. However, the quantitative impact of device pose and its variation on eye tracking applications running on such platforms has not been explored yet.

Method:

For deriving this metric, the eye tracking experiment is run on a tablet mounted with the tracker and the platform orientation (in roll, pitch yaw angles) is altered. Some of the tablet pose variations (variation in roll, pitch and yaw angles with respect to the neutral frontal position of the tablet where roll, pitch, yaw = 0) for the eye tracking study are shown in Figure 7a. The tablet is mounted on a tripod along with the eye tracker and its orientations are changed. Then the same UI and workflow as shown in Figure 1 is used with participants to collect gaze data from the tracker and gaze errors are estimated for each platform position. The platform motion sensitivity metric is shown below:

$$S_{pr} = \frac{\partial E_{pr}}{\partial P_r}, \; S_{pp} = \frac{\partial E_{pp}}{\partial P_p}, \; S_{py} = \frac{\partial E_{py}}{\partial P_y} \tag{15}$$

where S_{pr}, S_{pp} and S_{py} are the platform orientation sensitivities with respect to platform orientation roll, pitch and yaw angles respectively. P_r, P_p and P_y are the platform pose angles in roll, pitch and yaw directions and E_{pr}, E_{pp} and E_{py} are the corresponding gaze estimation errors (in degrees).



(**a**) (**b**)

Figure 7. (**a**) Shows the different orientations of the tablet mounted with the eye tracker in neutral position (top left) and (clockwise from top) roll, pitch and yaw variations of the tablet+ tracker setup. (**b**) Shows the gaze error sensitivity to orientations of the tablet when eye tracking is performed on it.

Example Results:

It is seen that for tablets, the gaze errors are most sensitive to pitch angle variations of the tablet + tracker pose (rotation about Y axis) while roll and yaw variations do not have appreciable effect.

4.3.3. Gaze Tracking Efficiency (Gaze Error Sensitivity to User Distance and Gaze Angle)

This metric is relevant to studying the impact of user distance and viewing angle on gaze error. Data from UD 45, UD60 and UD75 experiments are used to derive and test this metric.

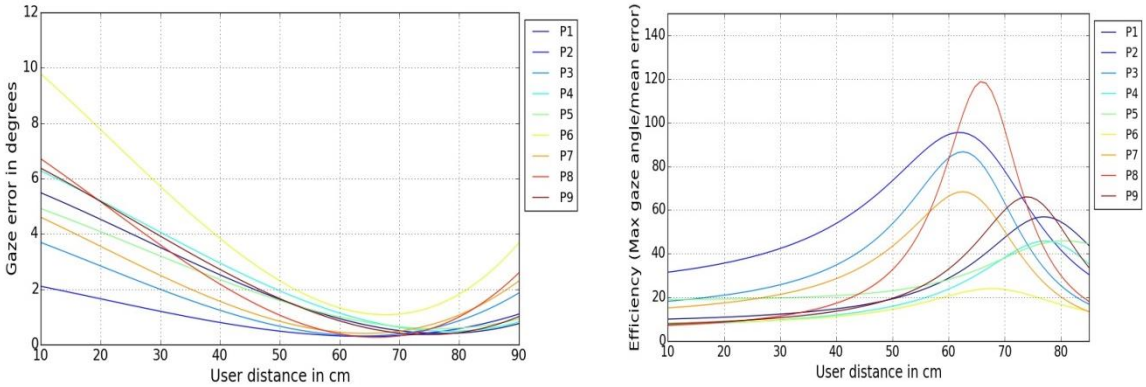
Method:

Gaze data was collected from 9 participants (named P1 to P9) at 3 different user tracker distances (45, 60, 75 cm) for each person. The gaze estimation error vs. user gaze angle is plotted in Figure 8a below. An efficiency metric essentially gives an idea about how to obtain maximum output for given range of inputs. In this case, for a certain user distance, the gaze tracking efficiency measure is given by:

$$(\text{Max gaze angle})_{\text{distance}} / (\text{mean error})_{\text{distance}} \tag{16}$$

The gaze angles are estimated by Equation (5) above and interpolated for the user distances. The mean error here is the averaged gaze estimation error for a particular user for a certain user-tracker distance. Gaze efficiency for the different user distances is plotted in Figure 8b below.

Example Results:



(**a**) (**b**)

Figure 8. (**a**) On the left shows the variation of gaze estimation error (estimated using Equation (8)) with user-tracker distance for 9 users (Each line shows error data for one user). (**b**) Shows the gaze tracking efficiency varying with user distance from tracker, estimated using Equation (16). (**b**) Shows that for most users, the tracking efficiency reaches a peak at a distance of 65–75 cm from the tracker. This indicates the best operating distance for the tracker at which gaze errors are minimum.

In a remote gaze tracking setup, it is not known where the users should be ideally positioned in front of the tracker to achieve best results. The gaze tracking efficiency metric defined here may be useful to quantitatively estimate for which user distance and viewing angles the best tracking accuracy can be obtained for a given tracker for a range of user-tracker distances. Figure 8 shows that gaze tracking errors reduce (and efficiency increases) as the user distance from the eye tracker increases until a certain value, after which tracking errors show an increasing trend, with the best gaze tracking performance achieved between the distances 65–70 cm for the given tracker. This can be explained, as above a user-tracker distance of 70 cm, the errors increase (and tracking is ultimately lost around 80 cm) because user eyes are poorly detected by the eye tracker cameras. The possible reason for having worse errors at closer distances for the remote eye tracker could be explained by the fact that at shorter distances or larger viewing angles the eye tracker may not detect the eye pupil center accurately because the eye rotation angles are large. Similar results have been reported in [48,49]. However, for head-mounted eye trackers that have a scene camera, the gaze estimation errors may actually increase with user distance due to the influence of parallax errors which happens due to the spatial offset between the eye and the scene camera [50].

It may be noted that the exact dependency of gaze tracking errors on user distance for every tracker may be different depending on the tracker design or components (e.g., camera quality), and so this metric needs to be evaluated quantitatively for each tracker to obtain their best performance.

4.3.4. Error Spatial Density (Sensitivity to Spatial Locations of Targets on the Display)

Error magnitudes or statistics do not provide any idea about the actual spatial dependence of gaze error values on the screen where a user's gaze is tracked. It cannot be understood if the error is uniform over the screen area or some parts of the screen are prone to more errors because of variability in the users' visual acuity or due to display properties of the screen being used. For this purpose, a spatial error density metric is derived to represent the area wise distribution of gaze errors on the screen.

Method:

The frontal display screen used during the experiments is divided into rectangular blocks around each AOI or stimulus target on screen. The error spatial density at each AOI is given by:

$$\left(\frac{\sum_{i=1}^{m} \frac{E_i}{m}}{(\mu * \text{number of x pixels}) * (\mu * \text{number of y pixels})} \right)_{\text{AOI}} \quad (17)$$

where, m is the number of gaze data points recorded by the tracker around each AOI (m ~90) while a user looks at them during an experiment session, E_i is the error at each data point estimated using Equation (8). The numerator therefore represents the mean error around each AOI (in degrees) and denominator is the total area of the rectangular block containing the AOI.

Example Results:

This metric is further discussed along with visualization in Section 5, where the results from this metric are depicted as a heat-map of gaze error densities plotted around each AOI on the entire monitor screen area.

4.4. Gaze Error Analysis with Respect to Visual Eccentricity

The location of gaze targets on the display screen may have a significant effect on the quality of gaze data obtained from an eye tracker. Visual performance of human eye is improved (fast and accurate) when a stimulus target is presented more centrally to the fovea, and worsens when the target is further in the periphery of the retina. Accordingly, a given tracker may show good accuracy in lower visual eccentricity areas and worse performance in high eccentricity areas. Therefore, measuring gaze error characteristics with respect to stimulus eccentricity is valuable in specifying the level of gaze errors under various operating conditions for a given tracker.

Method:

For this, our stimulus grid shown in Figure 2a was used to create an eccentricity map for the stimuli points and gaze error was analyzed separately for each eccentricity region. Figure 9a below shows our stimulus grid with its AOI (or stimuli) locations named AOI-1 to AOI-15. Figure 9b shows the rectangular stimulus grid superimposed with ellipses corresponding to the eccentricity of different stimuli locations. The parametric equation for ellipses centered around (0,0) may be stated as Equation (18) below, where a is the length of its semi-major axis and b is the length of its semi-minor axis and θ ranges from 0 to 2π:

$$x = a \cos(\theta), y = b \sin(\theta) \quad (18)$$

In our case, the two elliptical regions R1 and R2 in Figure 9b were obtained by converting AOI locations of the rectangular grid of Figure 9a to respective polar coordinates using Equation (18). Coordinates of AOI numbers 3, 7, 9, 13 form the central (dark blue) ellipse which corresponds to lowest visual angles while AOI numbers 6, 10 fall in the higher visual angle regions of the light blue ellipse. AOIs outside the larger blue ellipse (e.g., AOI numbers 1, 5, 11, 15) are regions of maximum visual eccentricity. This way the rectangular stimuli area is split into different regions corresponding to different visual eccentricity values and gaze errors in each of these areas are separately analyzed. The magnitudes of visual eccentricities for each region are shown through the colormap of Figure 9b, where dark colors represent a lower visual angle and vice versa. For studying the impact of visual eccentricity on gaze errors, data from UD45, UD 60 and UD 75 experiments are taken and an example gaze error map for the eccentricity regions is presented in Figure 9c below. Table 5 shows the results comprising of visual angles corresponding to three eccentricity regions of the screen and corresponding gaze error values for 4 participants, using data from three experiments (UD45, 60, 75).

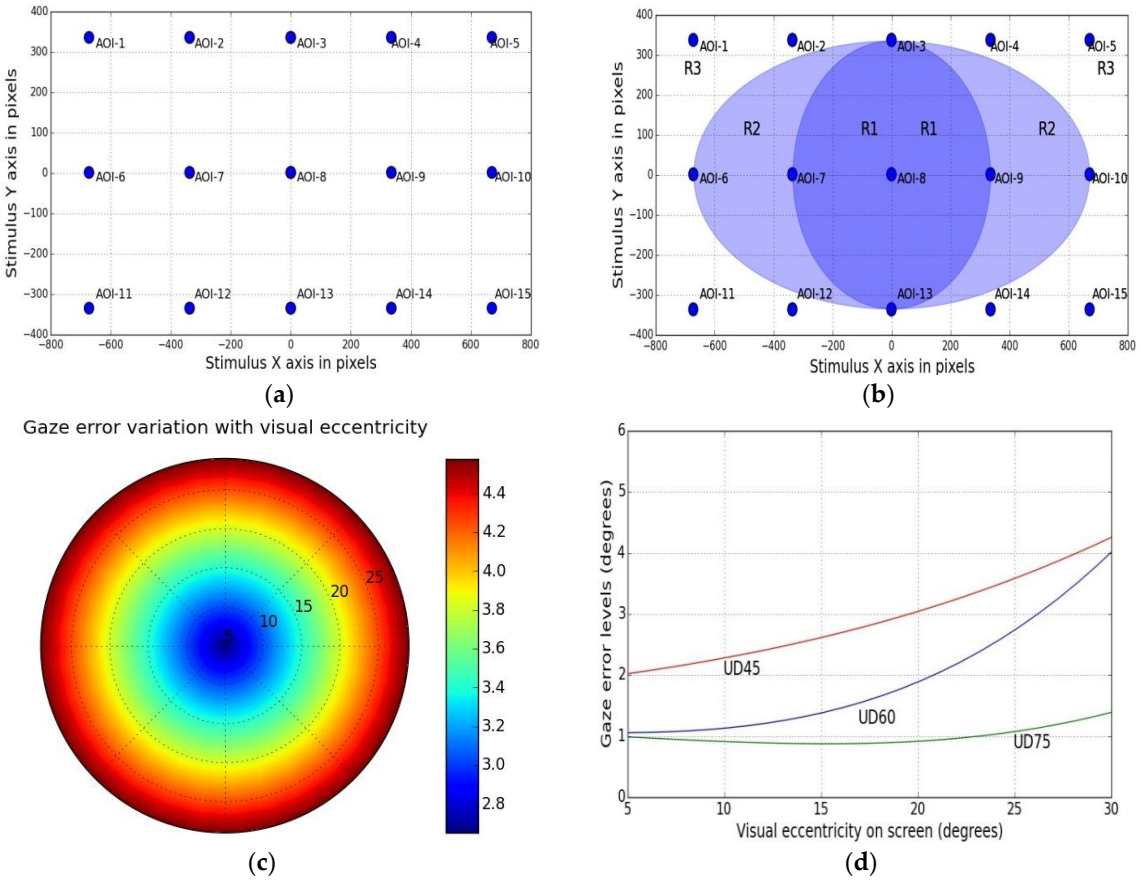


Figure 9. (**a**) The rectangular stimulus grid used in our experiments showing AOIs and AOI numbers. (**b**) Shows an eccentricity map constructed on the same stimulus grid, using data from UD75 experiments as an example. The dark blue inner ellipse region R1 has lowest visual eccentricity (11 degrees). The light blue larger ellipse region R2 has higher values (18 degrees) and the region outside R2 (region R3) has highest eccentricity (above 22 degrees). (**c**) This figure shows a polar plot of gaze error levels mapped with respect to visual eccentricity values (varying between 5 to 30 degrees) on the display screen, using data from UD45 experiment. It is seen that the gaze errors at the center are minimum and they increase to higher values with increasing visual eccentricities. The colorbar represents gaze error levels starting from low (blue) to high (red). (**d**) Shows the variation of gaze errors with visual eccentricity plotted using data from three user distances (45, 60, 75 cm). It is seen that when the user is close to the screen, gaze errors are more sensitive to visual eccentricities, whereas data for long user distances (e.g., UD 75) shows less sensitivity to eccentricities.

Example Results and Discussion

Table 5. Gaze error analysis with respect to visual eccentricity, data from 4 users for 3 user distances. Units of visual angles in columns 3, 5, 7 are in degrees.

Eccentricity Region	User. Number	Visual Angles at 45 cm	Mean Error for 45 cm (degrees)	Visual Angles at 60 cm	Mean Error for 60 cm (degrees)	Visual Angles at 75 cm	Mean Error for 75 cm (degrees)
R1							
	User1	15.47	2.65	12.5	1.28	10.04	1.06
	User2	20.32	2.19	12.99	0.8	9.98	1.12
	User3	15.95	2.16	13.31	1.27	10.14	0.96
	User4	15.8	2.32	12.75	1.04	10.19	0.91
R2							
	User1	25.15	3.6	20.55	1.54	16.91	0.96
	User2	26.39	2.37	21.01	1.08	16.90	0.90
	User3	24.83	3.9	20.18	1.91	17.16	0.71
	User4	25.19	3.57	20.13	1.96	16.99	0.88
R3							
	User1	32.10	4.58	26.08	2.14	21.70	1.15
	User2	32.72	3.96	29.46	1.24	21.94	0.91
	User3	32.27	4.41	25.40	2.82	21.69	1.16
	User4	32.02	4.66	25.34	2.88	21.90	0.96

Table 5 above shows gaze error results from our UD45-75 experiments for four users, when the stimulus AOIs are split into different visual eccentricity regions. The table shows several significant gaze error characteristics when gaze data is analyzed with respect to visual eccentricity. Firstly it is seen that for all user distances, the Region 3 has the highest visual angles and also correspond to the high error levels for any participant. Therefore Region 3 which comprises of screen corners is not suitable for reliable gaze tracking for the given tracker, whereas Region 1 (or central regions of low visual angles) are more reliable. Secondly, it is seen that gaze errors at 75 cm distance are low and mostly similar for all eccentricity regions for all users. On the other hand, errors at 45 and 60 cm distance are strongly sensitive to stimulus eccentricity. This has the implication that at longer user-tracker distances (75 cm), the entire screen area may be used for reliable gaze tracking, whereas if a user is positioned closer (e.g., less than 60 cm), errors may increase sharply with stimulus eccentricities, and so the entire screen area may not be usable. For such close user distances eye tracking has to be limited within certain low eccentricity angles (e.g., within regions 1 or 2).

The variation of gaze error with visual eccentricity is shown in Figure 9c above where gaze errors are interpolated and mapped to visual eccentricity values (starting with low values at the center to larger values outwards). This plot is created using UD45 data. Figure 9d shows the sensitivity of gaze error to visual eccentricity using data from three experiments (UD45, UD60, UD75). This plot signifies that for data collected above 60 cm, tracking accuracy is more uniform over the screen area, whereas for lower user distances, gaze errors are more sensitive to eccentricities. It may again be noted that these are example results and each tracker will have its own error characteristics with respect to eccentricities. Here the main aim is to present the eccentricity based tracker performance analysis procedure and demonstrate its significance using collected data.

4.5. The ROC Metric: Diagnostic or Subjective Performance Evaluation of Eye Tracking Systems

For in-depth performance analysis of an eye tracker using its data, the concept of a subjective performance evaluation metric rather than an objective one is introduced for the first time in this work. Objective or absolute performance measures like angular accuracy are independent of any prevalent criteria set by an observer while subjective performance measures may depend on specific accuracy

thresholds set by the observer (observer is the person evaluating the system). With respect to eye trackers, this means that the absolute accuracy values may not be sufficient to know if a tracker is good or not, as a tracker may have different performance at different accuracy thresholds.

For example, if a specific gaze application needs a minimum angular accuracy of 1 degree, a gaze tracker's performance may be termed "bad" if there are a lot of data points where errors are higher than 1 degree. However, for the same tracker, if the minimum required accuracy by the application (or the accuracy threshold) is set to be 2 degrees, the performance may be called "good" if there are a very few number of points where errors are higher than 2 degrees. In other words, the tracker can track reliably if the accuracy threshold is set to 2 degrees, but will produce mostly errors if desired accuracy level is set at 1 degree. Thus performance is subjective to the set error threshold. However, there needs to be a method to know how a tracker would perform if these desired accuracy thresholds are set and varied between high and low. Such a method is described below.

Method: The ROC (Receiver Operating Characteristic) Concept for Performance Evaluation

To estimate a tracker's performance with respect to different accuracy thresholds, the concept of ROC or Receiver Operating Characteristics [51,52] is implemented here using data from our gaze tracking experiments. Traditionally, ROC has been used in several cross-disciplinary fields like medicine [53] bio-informatics [54] and also computer vision for describing classification performance [55]. In this work, the ROC is used in determining the performance of our tracker when certain accuracy thresholds are set on its output data. This method proposed in this work is a new and experimental study and such an approach for evaluating eye tracking systems is different from any conventional analysis technique that has been done before. However, it opens up a novel way to look at performance characterization of eye trackers and helps to assess a tracker's usability.

Example Results: Implementation of Roc Concept for Subjective Evaluation

To estimate a tracker's performance with respect to different accuracy thresholds, the ROC concept is applied here. An ROC curve is plotted with the True Positive Rate (TPR) vs. False Positive Rate (FPR). In order to estimate the values of the TPR and FPR, one needs to estimate the number of true positives, true negatives, false positives and false negatives from the gaze tracker's data.

For estimating the true positives and negatives, we follow the analogy of estimating ROC for classifiers which need the Predicted values and Actual values. Starting with only raw error data for each gaze data point, we define the predicted values with respect to pre-set error threshold (ET) and actual values with respect to data mean M using the logic below:

The predicted value for a data point is positive if gaze error for that point > ET
The actual value for a data point is positive if gaze error for that point < M
The predicted value for a data point is negative if gaze error for that point < ET
The actual value for a data point is negative if gaze error for that point > M

The true positive (TP), true negative (TN), false positive (FP) and false negative (FN) values are therefore obtained as shown below in Table 6:

Table 6. Estimating the values of TP, TN, FP and FN using gaze error threshold.

Variables	Logic for Estimation
TP	Logical AND (Gaze error > ET, gaze error $\leq$ M)
FP	Logical AND (Gaze error > ET, gaze error > M)
TN	Logical AND (Gaze error < ET, gaze error > M)
FN	Logical AND (Gaze error < ET, gaze error $\leq$ M)

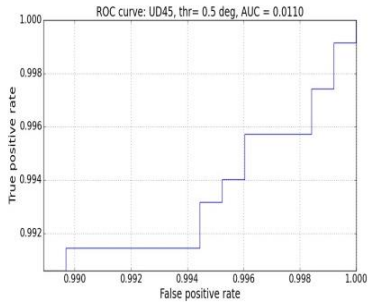
Then, for constructing an ROC curve, the TPR and FPR values are defined as:

$$\text{True Positive Rate } = \frac{\text{TP}}{\text{TP } + \text{ FN}}, \text{ False positive rate } = \frac{\text{FP}}{\text{FP } + \text{ TN}} \tag{19}$$

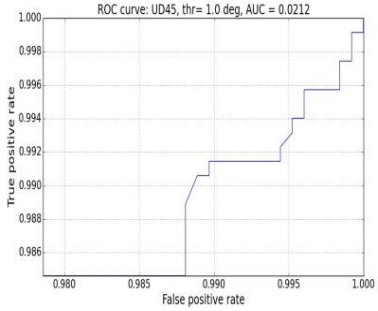
To validate the proposed subjective performance evaluation principle, these equations are used to plot the ROC curves below for six different preset threshold error values (ET) of 0.5, 1.0, 2.0, 3.0, 4.0, 5.0 degrees on each of two gaze datasets from UD45 and UD75 experiments. The plots below show the effect of setting respective error thresholds (ET) to estimate TP, TN, FP, FN values and TPR and FPR estimates using the logic described above, and the corresponding changes in characteristics of ROC curves are clearly visible in the subsequent plots.

Explanation and Analysis of Results

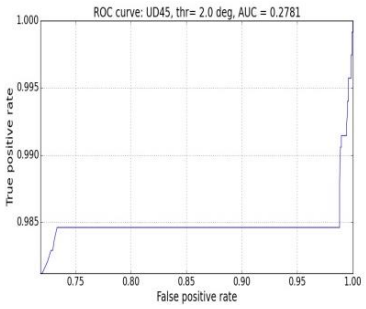
Several error threshold values between 0.5 to 5 degrees are applied to construct the ROC curves. The area under the ROC curve (called AUC) is computed for all the plots. It is seen that for UD45 data, highest AUC values are obtained at error thresholds between 3 to 5 degrees. But the highest AUC values for UD75 data are found between threshold values of 0.5–1.0 degree. The changing AUC values for the two datasets, as seen from the Figure 10a–l indicate that for the UD45 data the probability of achieving best performance lies between 3 to 5 degrees of gaze tracking accuracy, whereas for UD75 data, best achievable accuracy lies between 0.5–1.0 degrees. The threshold intervals can be made even shorter (e.g., 0.5 degrees) to study corresponding variation of AUC values and determine best obtainable accuracy from the eye tracker. Similar results are observed for data from 13 participants. It may be noted that the data statistics computed on these two datasets (UD45 and UD75) and presented in Sections 4.2.1 and 4.4 supports the ROC metric result. However, the benefit of the ROC technique is that users can set and observe the effect of varying error thresholds and determine how to obtain best performance from the tracker. They may also know if a certain tracker is suitable for them according to the required error threshold for a specific application.



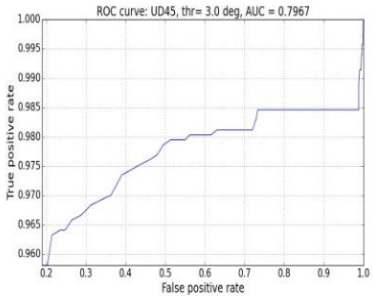
(**a**) UD = 45, ET = 0.5 deg, AUC = 0.01



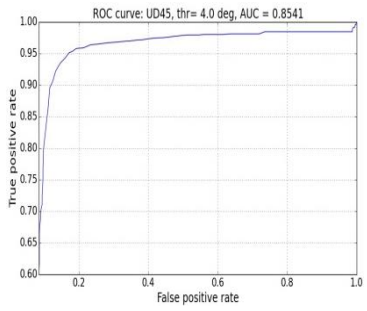
(**b**) ET = 1.0 deg, AUC = 0.02



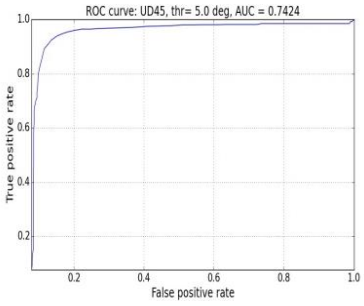
(**c**) ET = 2.0 deg, AUC = 0.27



(**d**) ET = 3.0 deg, AUC = 0.79

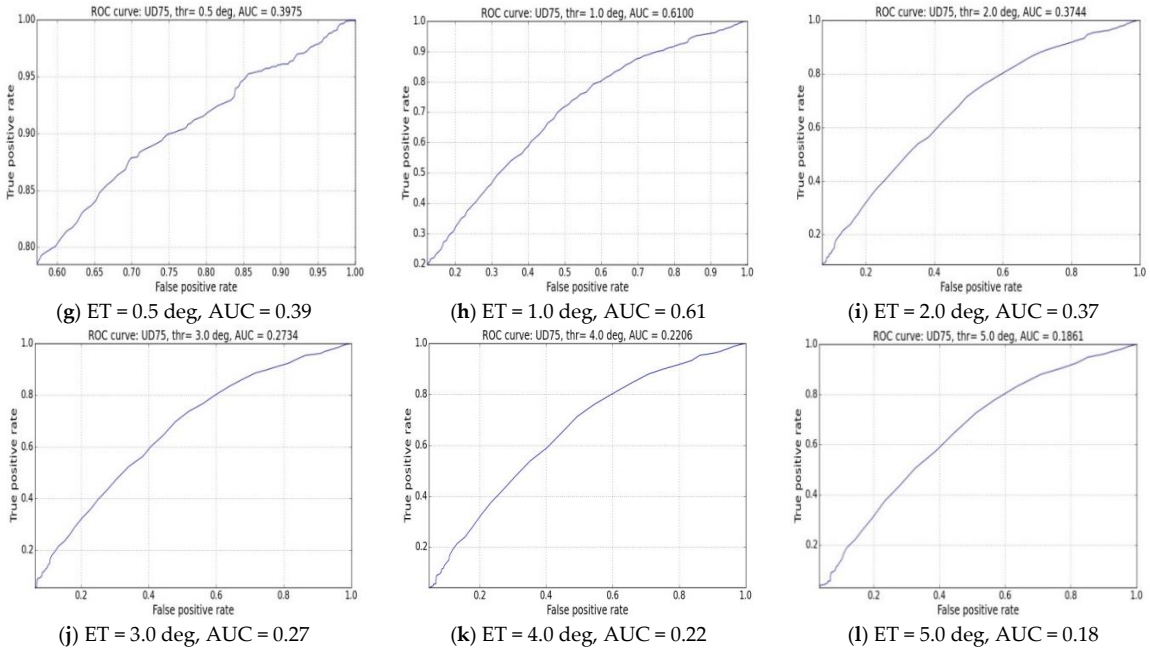


(**e**) ET = 4.0. deg, AUC = 0.85



(**f**) ET = 5.0 deg, AUC = 0.74

Figure 10. *Cont.*



(**g**) ET = 0.5 deg, AUC = 0.39 (**h**) ET = 1.0 deg, AUC = 0.61 (**i**) ET = 2.0 deg, AUC = 0.37

(**j**) ET = 3.0 deg, AUC = 0.27 (**k**) ET = 4.0 deg, AUC = 0.22 (**l**) ET = 5.0 deg, AUC = 0.18

Figure 10. ROC curves for one user, setting different thresholds for gaze datasets (**a–f**) UD45 and (**g–l**) UD75.

4.6. Discussion

This section introduced and defined several new gaze data metrics and analysis procedures that may reflect the all-round performance of a gaze estimation system by inspecting its data in great details to reveal patterns that are not visible by just looking at simple average accuracy measures. The methods described in this section use only the raw eye gaze data (x, y coordinates of tracked gaze locations) and ground truth (AOI locations on the UI) and therefore are independent of the tracking system or algorithm. This type of analytical methods may be useful to all kinds of gaze research and developmental works for complete evaluation of a gaze tracker's system behavior and understanding of system capabilities and limits. The codes for the implementing the methods and metrics will be published in the project repository (https://github.com/anuradhakar49/GazeVisual) for use by the eye gaze research community in the coming months.

A few points regarding applicability of the metrics described above need to be discussed. In this work, the derivations of the gaze accuracy metrics have been done using data from both left and right eye. However, there may be eye tracking systems and gaze data collection software which provide only monocular data, i.e., data from one eye only. In these cases, the metrics will still work on the monocular gaze data, but the results might be different than that from using both eye data.

Another aspect is that there may be different tracker intrinsic parameters which affect eye tracking data quality and accuracy. These include choice of fixation detection settings, noise reduction and averaging window parameters to name a few. However, in this work we focused on studying the impact of extrinsic conditions on the tracker data characteristics, which arise mainly from changes in the operating environment of the tracker. The reason is firstly that impact of these operating conditions are sparsely investigated and quantitatively evaluated in eye gaze research. Secondly, there are commercial eye trackers which do not allow access to their device software or data parser settings and therefore there is no way to study the impact of intrinsic parameters on their data. Therefore, in order to propose a set of evaluation methods that can be applicable to any kind of eye tracker, the scope of this work is limited to studying tracker independent factors only.

Apart from defining quantitative metrics, the aspect of gaze error sensitivity to visual eccentricity is studied in Section 4.4, as it may have significant implications on the level of gaze accuracy obtainable

from a tracker. A tracker may show low error levels at low visual eccentricities and high errors when eccentricities increase. The analysis method in Section 4.4 was therefore described which may be useful for eye tracking researchers as a basis for comparison of eye trackers as well as for determining tolerable eccentricities for a tracker under various operating conditions and also for designing proper stimuli for eye tracking applications.

5. Visualizations for Evaluating Gaze Estimation Systems

Graphical tools are key to viewing large volumes of gaze data and understanding data characteristics like error magnitude, bias and impact of variable operating conditions [29,56]. Researchers commonly design their individual test routines and report occurrence of errors mostly in statistical or tabular format. While software packages exist for gaze data analysis in some interdisciplinary areas like psychology, perception and usability research [57], there are no standard performance visualization/analysis tools available for researchers or developers working on gaze tracking algorithm/applications. Below, therefore some visualization methods are presented which may be used to study the accuracy of a generic gaze tracking system under variable operating conditions and estimate its overall performance. These visualization concepts are inspired from research fields like data science, visual analytics and statistics [58–60] and are implemented on data collected from our user distance, head-pose and platform movement experiments done using the test UI and workflow as shown in Figure 2a,b. The visualizations used in gaze research till now, as described in Section 2 are mostly focused on relating eye tracking data to human cognitive aspects (like attention, search patterns and interest) in the form of heat maps and fixation maps. However, dedicated visualizations for eye tracking performance analysis using raw gaze data from a tracker under test have not been developed in this much detail before.

The visualizations presented in this work are divided into three subsections. The 1st subsection (A) comprises of methods for visualizing data from individual data files (e.g., files containing data of a single user or on a single experimental variable). The 2nd subsection (B) comprises of methods for aggregating and visualizing data (and its various properties) from multiple files (e.g., files containing data from multiple experiments or from multiple users or both). The 3rd subsection (C) presents the concept of the graphical software interface named GazeVisual which is being currently developed for easily accessing the gaze data metrics and visualization tools. As done in the previous Section 4 each visualization sub-subsection is split into "Method" describing the concept and "Results" describing the outcome and significance of the visuals tested on data from our different experiments.

5.1. Plots for Data Visualization from Single Eye Tracking Experiments

Subcategories in this include point and spatial error density maps, plotting of gaze angular field in the tracking space and 3D error magnitude plots. Special visualizations have been designed to represent impacts of head pose and platform orientation on gaze error levels.

5.1.1. Data Density Maps

Using data density maps, raw gaze data (gaze x, y coordinates collected at each target location of UI shown in Figure 1a) can be plotted as data point clusters, color-mapped according to point densities (calculated using histograms) around each AOI.

Method:

Gaze errors are estimated using Equations (1)–(8) on data from UD45 experiment. This is then used to plot a color-mapped density plot from one participant and shown in Figure 11a.

Example Results:

Figure 11a shows the gaze data point density around each target location on the overall screen area (data from UD45 experiment), with the scale alongside showing number of data points mapped

to colors. This type of plots give a quick look into the relative scatter of data points at each AOI with a number scale alongside, which makes it easier to interpret the data patterns and detect any anomaly.

5.1.2. Spatial Error Heat Maps

The concept of gaze spatial error density was discussed in Section 4.3.4, where the formula for gaze error density around each AOI on the display screen (Equation (17)) was derived.

Method:

For visualizing the spatial distribution of errors on the display screen, a plot with gaze error densities around AOIs (using Equation (17) and data from UD60 experiment) color-mapped with values is shown in Figure 11b. The scale represents the color-map of error density values in degree/cm^2.

Example Results:

The gaze spatial error analysis heatmap can help to identify whether the gaze tracking accuracy is uniform over the monitor display and detect most probable locations for error on the screen. For example, from our study, the top left corners of the monitor were found to be prone to higher error values for all participants. This type of plots may help eye tracker users to improve gaze tracking performance, for example by checking the display screen quality, eye tracker mounting issues or compensating for the tracker's performance at certain gaze angles if non-uniform or anomalous error densities are observed. It may be noted that error heat-maps are different from data density maps of Figure 11a, as the heat-maps take into account the display properties (screen size and resolution) of the computer screen where gaze is tracked.

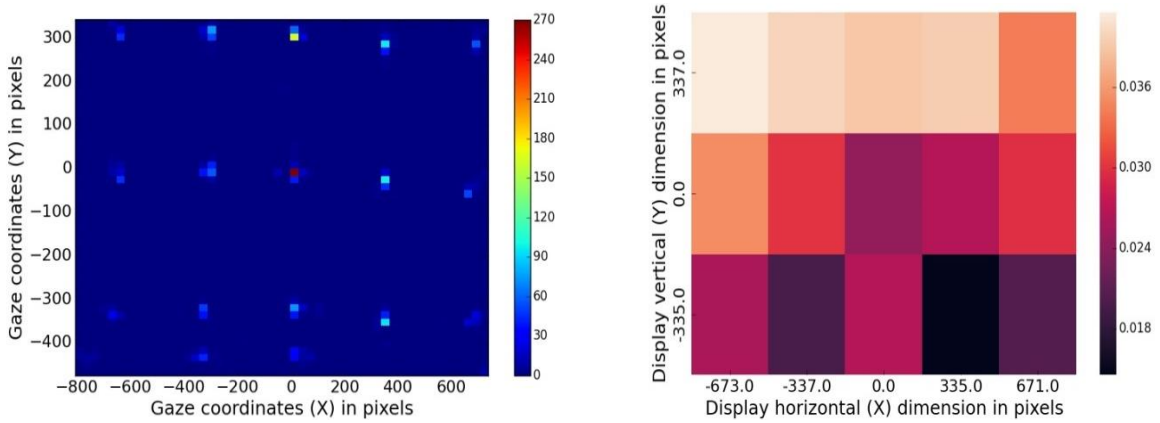


Figure 11. (**a**) Scatter density plot of gaze errors. (**b**) Shows a spatial error heat map for all AOI locations on the display screen (coordinate of screen center is 0,0) constructed from UD75 data. The gaze error density at all AOI locations has the units of degree/cm^2.

5.1.3. Gaze Error vs. Visual Field

The visual field describes the extent of the observable area that is seen at any given moment. The UD45-75 experiments revealed dependence of gaze error values on user-tracker distances and user visual angles quantitatively, as discussed in Section 4.3.3.

Method:

The variation of the visual angle with user distance from the tracker is shown as angular sectors in Figure 12a which also shows the dependence of gaze estimation accuracy on the user visual field. The tracker position with respect to the user is marked as tracker distance or "TD" in the plot. The magnitude of gaze estimation errors (derived from UD45-75 experiment data) for each visual

angle is represented by the color intensity of the sectors. The corresponding color-map of gaze errors (in which low intensity of color means lower error) is shown alongside.

Example Results:

This visualization of Figure 12a shows several interesting features, firstly the reduction in user viewing angle with increasing user distance from tracker (shown by narrowing sectors). Secondly, it shows that the given tracker achieves best accuracy at a narrow visual angle of less than 20 degrees (compared to normal human field of view which is about 60 degrees on either side of frontal eye position). It also shows that gaze errors increase significantly with even slight increase in user visual angles (demonstrating the tracker's sensitivity to user distance). These kinds of plots, along with the visual eccentricity plots described in Section 4.4 (Figure 9c) can help eye tracking researchers to know where to position a user in front of an eye tracker for best tracking results.

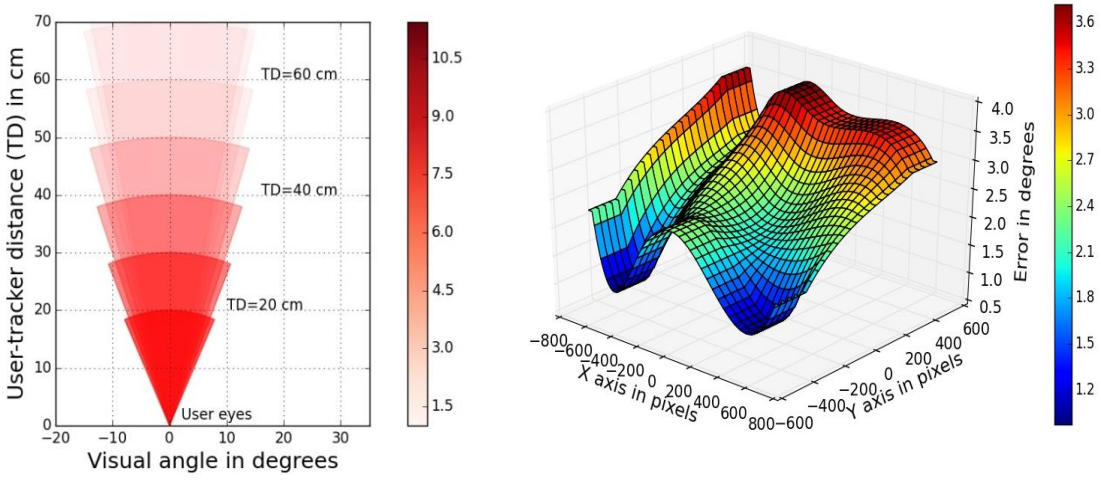
5.1.4. 3D Gaze Error Distribution Plot

Method:

The 3D gaze error distribution plot shows the magnitude of gaze errors as a function of X and Y dimensions of the viewing area on the display screen. The gaze errors are plotted along Z axis and X and Y axis of the plot represent horizontal and vertical dimensions of the display in pixels, as shown in Figure 12b. Data from UD75 experiment for one user is used for the plot. The scale alongside shows the color-mapped magnitudes of gaze error values over the display screen area.

Example Results:

These plots help to diagnose gaze error levels over the display area. For example high error values are found to occur near the display corners and near the screen borders which could be due to high visual angles in those regions. Impact of display screen locations where gaze is tracked on corresponding gaze error levels has been reported in [61] which shows that gaze errors may increase by 33% on the screen border regions compared to those in the central regions.



(**a**) (**b**)

Figure 12. The sectors in the plot (**a**) on the left represent the visual angles of a user in degrees and are color-mapped according to gaze error obtained for each visual angle. The color-bar alongside shows the mapping to the magnitude of gaze errors in degrees. (**b**) Shows a 3D plot of gaze errors as a function of display X, Y dimensions (in pixels) for data from UD45 experiment along with error magnitudes mapped in a color bar alongside.

5.1.5. Platform Movement Tolerance Conics

A set of conics are used here to represent the degrees of freedom of movement "allowed" in a handheld device about its central axis if eye tracking is to be done on it with sufficient accuracy. The tablet angular pose variations in each of roll, pitch yaw directions, and corresponding gaze error values are derived from platform orientation sensitivity experiments as described in Section 4.3.2.

Method:

Data from the platform orientation experiments are used to plot the conics in Figure 13a. The cones represent the range of platform angular pose variations (in roll, pitch, yaw directions), which occur due to variable hand poses when the tablet is held by a generic user. The aperture (or vertex angle) of each cone represents the maximum angle to which the edges of the device can be tilted with respect to the frontal position (where roll, pitch, yaw = 0) without gaze errors exceeding 1 degree.

Example Results:

As observed from the platform orientation sensitivity curves in Figure 7b, the vertex angle of the cone corresponding to platform pitch movement is the smallest and about 5 degrees while the cone angle is about 10 degrees in the other directions of tablet roll and yaw. The utility of this visualization is that it shows the practical movement limits and maximum platform pose variations that are allowed for a certain tablet-tracker setup to perform reliable eye tracking on it.

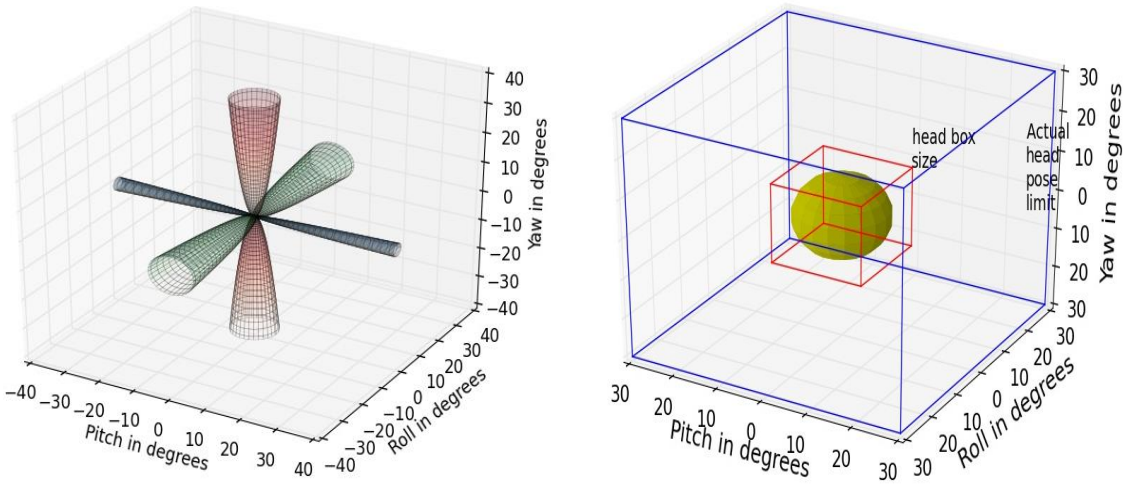
5.1.6. Head-Pose Tolerance Box

Method:

This tolerance box visualization (Figure 13b) shows the degree of head pose variations allowed by the given eye tracker while maintaining acceptable accuracy levels and is implemented using data from our head-pose experiments as described in Sections 2 and 4.3.1.

Example Results:

In the visualization of Figure 13b, the larger box shows the maximum degree of head movement (in roll-pitch-yaw angles) possible by an average user (whose head is shown by the yellow sphere) which is in the order of 30 degrees of angular movement in each direction of roll-pitch-yaw. The plot also shows that for reliable gaze estimation with the given tracker, the head pose of a user must be limited within the smaller "box", the dimensions of which are about 10 degrees in each angular direction and correspond to gaze error values of 0.5 degrees. Such plots constructed for eye trackers may help to understand their head-movement tolerances quantitatively.



(**a**) (**b**)

Figure 13. (**a**) Shows the freedom of movement of a dynamic platform like a tablet if one intends to perform eye tracking on it with sufficient accuracy. As seen from our experiments, gaze errors increase sharply with variations in the platform pose pitch angles and so the pitch movement of the tablet + eye tracker setup has to be kept within tight limits. As the impacts of yaw and roll variations are not very significant, larger pose variations is possible in those directions. (**b**) Shows the limits imposed by eye tracking on a desktop platform on head movement of a participant which has to be constrained within the small red box space to achieve reliable tracking accuracy.

5.2. Plots for Data Aggregation from Multiple Eye Tracking Experiments

These plots are useful in clustering results from the multiple eye tracking experiments to derive inferences about gaze error patterns and dependence on various experimental variables, conditions etc.

5.2.1. Stacked Multi-Graphs

These kinds of plots could be used for displaying multiple parameters from several participants from one or more experiments on a single plot [62]. These reduce the need for creating a large number of separate plots and make it easier for a viewer to retrieve, understand and compare information obtained from multiple users.

Method:

A stacked horizontal bar chart is constructed from our user distance experiments and shown in Figure 14a. In this, the mean gaze error and maximum gaze angles for three different distances for 13 different users (participant ID 1-13) are plotted together. The three colored bars represent three distances, the black bars' length shows mean error levels for each user, and the length of the colored bars' stands for the maximum gaze angle for each user.

Example Results:

This kind of plot helps to gain a quick look into a large gaze dataset obtained from any gaze experiment done with multiple users and draw inferences about the person-to-person variations of gaze data quality.

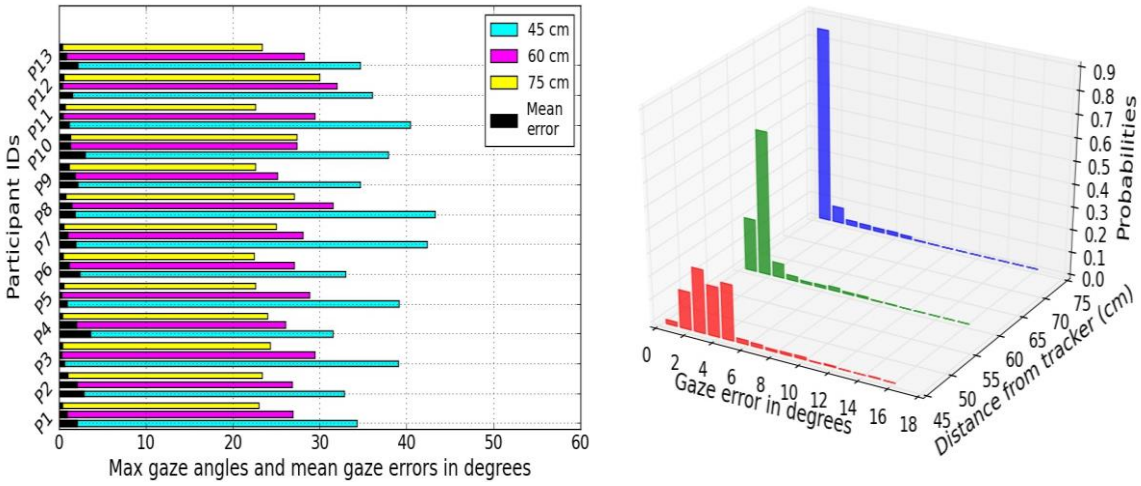


Figure 14. The figure on the left (**a**) shows gaze error and maximum gaze angle data from multiple experiments (UD45-75) for 13 participants (participant ID P1-P13) plotted together for easy visualization and comparison of results. (**b**) Shows gaze error data distributions from UD45-75 experiments plotted as 3D stacked histograms.

5.2.2. Stacked Distributions

For understanding data patterns, histograms provide more insight into data characteristics comparable to numerical values or statistics.

Method:

Histograms are constructed using gaze error data from the experiments UD45, UD60 and UD75, and plotted together as stacked 3D histograms in Figure 14b. These histograms offer information on where error values are concentrated, presence of data extremes or unusual data patterns.

Example Results:

The Figure 14b displays binned error values (number of bins = 20) for three different user distances and shows that with increasing distance, the errors move closer to lower error magnitudes. Stacked histograms plotted together on data collected from different experiments done under different conditions helps to see differences in gaze error patterns clearly and fast. These visuals could be useful to study impact of multiple experimental conditions on gaze errors from one person.

5.2.3. Angular Error Charts for Multiple Experimental Variables

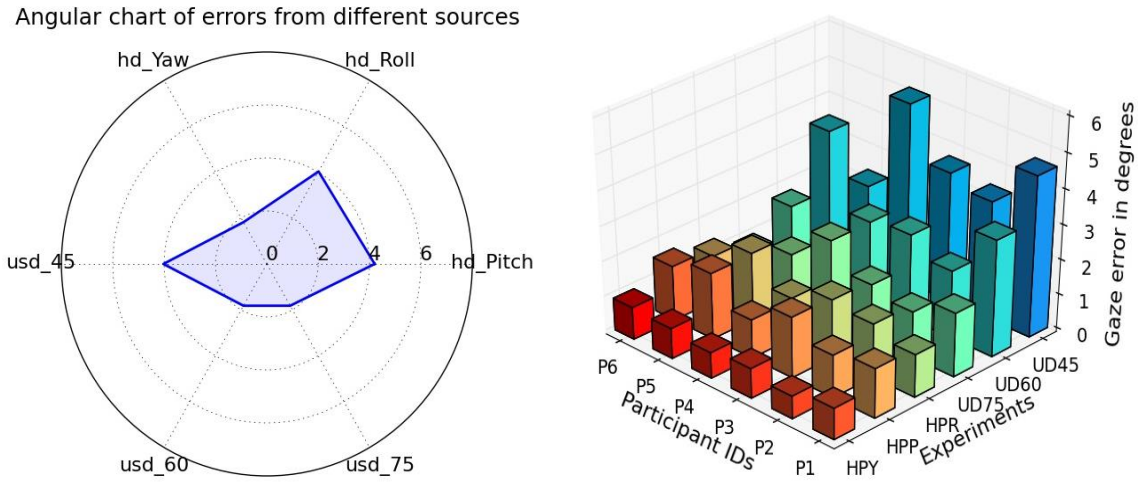
An angular chart can act as a powerful visualization method for plotting multivariate data while reporting performance of eye trackers [63]. When there is more than one factor, such as user distance, head pose angles that one needs to measure and compare while reporting tracker performance, this type of charts can come handy. In these charts, each variable under consideration is plotted on an axis that radiates out from a point (center value zero), with equal increments along each axis. These charts provide a compelling way of looking at data than simple tabular representations.

Method:

Angular charts are implemented here to compare multiple variables from a number of our gaze experiments quantitatively, as shown in (Figure15a), and the aim is to make it easy to identify which experimental variables result in higher errors.

Example Results:

This chart helps to understand the relative impact of factors like head movement, user distance etc. on the final gaze accuracy of the tracker. Each error source (e.g., each head pose direction or user distance) is treated as a "variable" plotted on an axis starting from the center (representing units of gaze angular accuracy) and all the variables are connected together to form a polygon [64]. All axes are equidistant and maintain same scale of angular accuracy on them. For example, from Figure 15a which aggregates data from our desktop based experiments, one can get the idea that the major error contributions in gaze error come from head roll and low user distances out of all other factors.



(**a**) (**b**)

Figure 15. The figure (**a**) on the left shows the relative magnitudes of mean gaze error from different experiments for one person plotted on an angular chart. (**b**) On the right shows a very compact representation of data from all our experiments for 6 participants clustered into a single plot. HPY, HPP and HPR refer to head pose yaw pitch and roll experiments respectively.

5.2.4. 3D Bar Clusters

Method:

This type of plot (Figure 15b) is implemented to aggregate data from multiple persons and all of our desktop based experiments to study and compare the gaze error levels of individuals across different experimental sessions. In the plot, different experiments are coded by colors with respective error values represented by bar heights, plotted along the Z axis.

Example Results:

This kind of plot can be useful in inspecting data from a large number of experiments, done with multiple participants and judge overall eye tracking performance by gaze error levels which is shown along the Z axis. For example, the plot 15(b) helps to draw several meaningful inferences about our multiple experiments. It shows that minimum error levels occur from head pause yaw variation experiments and highest ones from UD45 experiments for majority of participants. Also it is seen that participant P2 has lower errors in all experiments compared to other participants.

5.3. GazeVisual v1.1: Concept of a Performance Evaluation Tool for Eye Trackers

This work is currently underway in which the metrics and visualizations presented in this paper are being packaged into a composite graphical user interface or GUI [65] format named GazeVisual v1.1, and a snapshot of its current look and operational features is shown in Figure 16 below. Using this GUI, a user will be able to browse and load the specific gaze data files along with ground truth data,

compute and display results of various metrics (described in Sections 4 and 5) by clicking on the buttons and see the visualizations in the plot window.

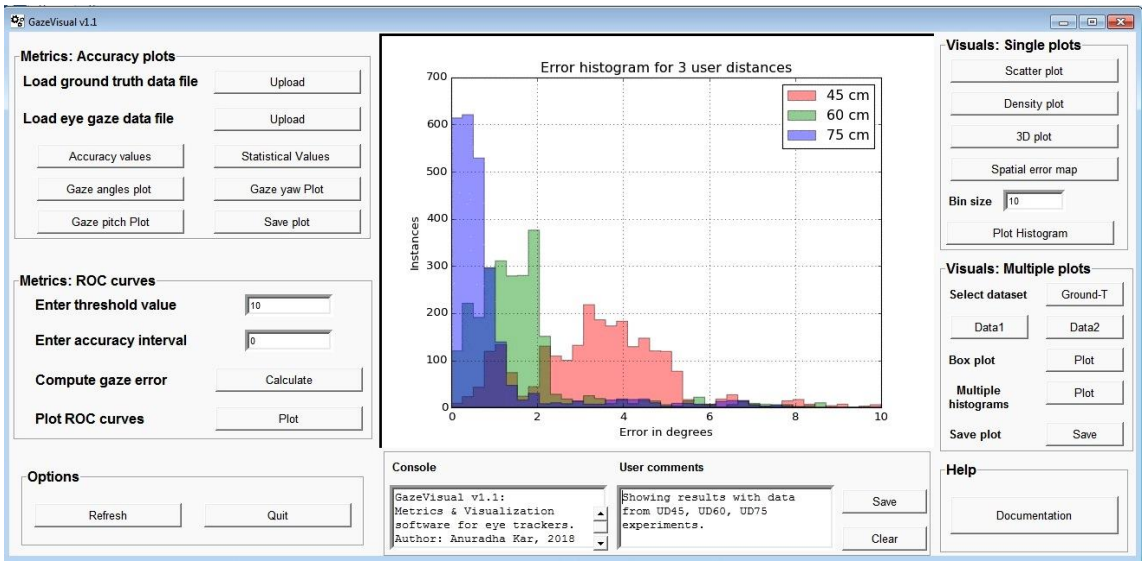


Figure 16. A snapshot of the graphical user interface GazeVisual v1.1 which is currently being developed for implementing the accuracy metrics and visualization concepts as discussed in Sections 4 and 5 above. A sample histogram plot using data from UD45, UD60 and UD75 experiments is displayed in the plot area of the GUI. Other visualizations are also to be shown in this area and the plots can be saved. A demo video of the software demonstrating a set of its functionalities may be found in the GitHub link: https://github.com/anuradhakar49/GazeVisual/tree/Demo-videos.

5.4. Discussion

In this section, different methods for visualization of eye gaze data from several experiments with an eye tracker are presented. These visual analysis techniques would enable effective inspection of data characteristics of a gaze tracker and means for studying error pattern variations when any operating condition, such as user distance, screen properties, head movements is altered. Overall these visualizations are designed to aid in the complete characterization as well as comparison of multiple eye trackers. It may be noted that the only ingredient needed to implement these plots are gaze error magnitudes computed using Equations (1)–(8) from raw gaze and ground truth data and some experimental variables like user distance and display properties. This makes these visuals easily adaptable for any eye tracking system using its data samples and ground truth.

It may however be noted that this paper does not go deeper into the gaze data characteristics of the used eye tracker itself but only uses the collected gaze datasets to demonstrate the implementation of proposed visual techniques. The reasons for it are explained at the end of the Introduction section where we define the purpose and scope for our work.

The visualization concepts presented in this section are inspired from interdisciplinary areas like data mining, image processing and visual analytics. As per our knowledge, there hasn't been any detailed work in gaze research so far for developing visualizations for studying data quality (especially to gain insight into gaze error levels, its distributions and spatial patterns) from generic eye trackers. The visualizations in this paper and the graphical software interface GazeVisual aids in this direction as also there is currently no publicly available software for performance evaluation of eye trackers. The implementation resources of the visualizations will be available in the open repository of the project (https://github.com/anuradhakar49/GazeVisual), for use and further development by the eye gaze community. The overall aim is to encourage eye gaze researchers to adopt more detailed ways of analyzing their eye tracker characteristics and providing open source tools and software for the same.

6. Conclusions

There is a strong need for standard open source tools and common methods for measurement and data analysis in eye gaze research, without which each researcher has to develop their own methods or rely on expensive software which does not allow customizations. This leads to presence of inadequate details in gaze research results and increased diversity in reporting formats, making the interpretation of research results difficult and cross-validation of outcomes nearly impossible. There also needs to be an agreement in the metrics used for reporting gaze accuracy and standard methods for complete characterization of eye trackers under different operating conditions.

Keeping these aspects in mind, in this paper, it is shown how the all-round performance of generic gaze tracking systems can be evaluated and compared numerically and visually. Several measures and visual methods for meaningful understanding of a gaze tracking system's behavior, solely from its data outputs are presented. Some of these metrics are linked to visualizations while some are independent and vice versa, which shows that metrics and visuals are tightly related and both are essential for complete system description of an eye tracker. These methods could be useful for any researcher or developer of eye gaze systems to estimate and validate the performance of their system under widely variable influencing conditions. Especially, these evaluation tests may help to study the robustness of an eye tracker and degree of change in its data quality quantitatively, when operating under unpredictable and harsh conditions in unconstrained eye tracking applications. Also these can better describe the reliability of a newly designed system rather than error values expressed in simple numerical formats. Overall the presented metrics and visualizations are expected to help researchers as well as common users in understanding and improving the all-round performance of generic gaze trackers.

The GazeVisual v1.1 interface is an open source graphical tool which is being designed on the model of standard data visualization software but till date no such publicly available software for performance evaluation of gaze trackers exist. This interface would make the evaluation methods described in this paper extremely easy to use for a general user, who simply needs to load their gaze and ground truth data to the software to access all the evaluation metrics and visualizations through a few button clicks.

The resources for implementing the concepts described in this paper will be released in an open repository where users can download these tools and modify them according to their own requirements and provide feedback for their improvement. The software components will be tested and accompany proper documentation to ensure seamless operation and easy adaptation by researchers. As future work in this direction, it is intended to include more system and external parameters into the evaluation criteria and study more user platforms for eye gaze applications such as eye trackers for large displays and head-mounted devices.

Author Contributions: A.K. conceived and designed the experiments, prepared the gaze tracking setups and ran the experiments for the collection of data. A.K. wrote the draft, and both authors discussed the contents of the manuscript. P.C. contributed to the research idea, did supervision of the work, provided feedback on the work, corrected the draft and approved the final version. Conceptualization, A.K.; Data curation, A.K.; Formal analysis, A.K.; Funding acquisition, P.C.; Investigation, A.K.; Methodology, A.K.; Project administration, P.C.; Software, A.K.; Supervision, P.C.; Visualization, A.K.; Writing—original draft, A.K.; Writing—review & editing, P.C.

Funding: The research work presented here was funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded by FotoNation Ltd. Project ID: 13/SPP/I2868 on "Next Generation Imaging for Smartphone and Embedded Platforms".

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kar, A.; Corcoran, P. A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms. *IEEE Access* **2017**, *5*, 16495–16519. [CrossRef]
2. Bulling, A.; Alt, F.; Schmidt, A. Increasing the security of gaze-based cued-recall graphical passwords using saliency masks. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12), Austin, TX, USA, 5–10 May 2012; ACM: New York, NY, USA, 2012; pp. 3011–3020.
3. Wood, E.; Bulling, A. EyeTab: Model-based gaze estimation on unmodified tablet computers. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14), Safety Harbor, FL, USA, 26–28 March 2014; ACM: New York, NY, USA, 2014; pp. 207–210. [CrossRef]
4. Kern, D.; Mahr, A.; Castronovo, S.; Schmidt, A.; Müller, C. Making use of drivers' glances onto the screen for explicit gaze-based interaction. In Proceedings of the 2nd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '10), Pittsburgh, PA, USA, 11–12 November 2010; ACM: New York, NY, USA, 2010; pp. 110–116.
5. Meissner, M.; Pfeiffer, J.; Pfeiffer, T.; Oppewal, H. Combining Virtual Reality and Mobile Eye Tracking to Provide a Naturalistic Experimental Environment for Shopper Research. *J. Bus. Res.* **2018**. [CrossRef]
6. Corcoran, P.M.; Nanu, F.; Petrescu, S.; Bigioi, P. Real-time eye gaze tracking for gaming design and consumer electronics systems. *IEEE Trans. Consum. Electron.* **2012**, *58*, 347–355. [CrossRef]
7. Morimoto, C.H.; Mimica, M.R.M. Eye gaze tracking techniques for interactive applications. *Comput. Vis. Image Underst.* **2005**, *98*, 4–24. [CrossRef]
8. Lee, H.C.; Luong, D.T.; Cho, C.W.; Lee, E.C.; Park, K.R. Gaze tracking system at a distance for controlling IPTV. *IEEE Trans. Consum. Electron.* **2010**, *56*, 2577–2583. [CrossRef]
9. Pfeiffer, J.; Pfeiffer, T.; Greif-Winzrieth, A.; Meissner, M.; Renner, P.; Weinhardt, C. Adapting Human-Computer-Interaction of Attentive Smart Glasses to the Trade-Off Conflict in Purchase Decisions: An Experiment in a Virtual Supermarket. In Proceedings of the International Conference on Augmented Cognition: Neurocognition and Machine Learning, Vancouver, BC, Canada, 9–14 July 2017; pp. 219–235.
10. Pfeiffer, J.; Meißner, M.; Prosiegel, J.; Pfeiffer, T. Classification of goal-directed search and exploratory search using mobile eye-tracking. In Proceedings of the International Conference on Information Systems (ICIS 2014), Auckland, New Zealand, 14–17 December 2014.
11. Hansen, D.W.; Ji, Q. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 478–500. [CrossRef] [PubMed]
12. Ruhland, K.; Peters, C.E.; Andrist, S.; Badler, J.B.; Badler, N.I.; Gleicher, M.; Mutlu, B.; McDonnell, R. A review of eye gaze in virtual agents social robotics and HCI: Behaviour generation user interaction and perception. *Comput. Gr. Forum* **2015**. [CrossRef]
13. Holmqvist, K.; Nyström, M.; Mulvey, F. Eye tracker data quality: What it is and how to measure it. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12), Santa Barbara, CA, USA, 28–30 March 2012; ACM: New York, NY, USA, 2012; pp. 45–52.
14. Reingold, E.M. Eye Tracking Research and Technology: Towards Objective Measurement of Data Quality. *Vis. Cognit.* **2014**, *22*, 635–652. [CrossRef] [PubMed]
15. Evans, K.M.; Jacobs, R.A.; Tarduno, J.A.; Pelz, J.B. Collecting and Analyzing Eye-Tracking Data in Outdoor Environments. *J. Eye Mov. Res.* **2012**, *5*, 1–19.
16. Špakov, O. Defining Standard Gaze Tracking API. In Proceedings of the CHI 2013 Workshop on "Gaze Interaction in the Post-WIMP World", Paris, France, 27 April–2 May 2013; pp. 1–4.
17. Kar, A.; Corcoran, P. Towards the development of a standardized performance evaluation framework for eye gaze estimation systems in consumer platforms. In Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 9–12 October 2016; pp. 2061–2066.
18. Mora, K.A.F.; Monay, F.; Odobez, J. Eyediap: A Database for the Development and Evaluation of Gaze Estimation Algorithms from RGB and RGB-D Cameras. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14), Safety Harbor, FL, USA, 26–28 March 2014; ACM: New York, NY, USA, 2014; pp. 255–258.
19. Pfeiffer, T. Towards Gaze Interaction in Immersive Virtual Reality: Evaluation of a Monocular Eye Tracking Set-Up. In *Virtuelle und Erweiterte RealitatFunfter Work. der Gifachgr*; Shaker Verlag GmbH: Aachen, Germany, 2008; pp. 81–92.

20. Ooms, K.; Lapon, L.; Dupont, L.; Popelka, S. Accuracy and precision of fixation locations recorded with the low-cost Eye Tribe tracker in different experimental set-ups. *J. Eye Mov. Res.* **2015**, *8*, 1–24.
21. Popelka, S.; Stachoň, Z.; Šašinka, Č.; Doležalová, J. EyeTribe tracker data accuracy evaluation and its interconnection with hypothesis software for cartographic purposes. *Comput. Intell. Neurosci.* **2016**, *2016*. [CrossRef] [PubMed]
22. Funke, G.; Greenlee, E.; Carter, M.; Dukes, A.; Brown, R.; Menke, L. Which eye tracker is right for your research? Performance evaluation of several cost variant eye trackers. In Proceedings of the Human Factors and Ergonomics Society 2016 Annual Meeting, Washington, DC, USA, 19–23 September 2016; pp. 1239–1243.
23. Mannaru, P.; Balasingam, B.; Pattipati, K.; Sibley, C.; Coyne, J.T. Performance Evaluation of the Gazepoint GP3 Eye Tracking Device Based on Pupil Dilation. In *Augmented Cognition: Neurocognition and Machine Learning*; Schmorrow, D., Fidopiastis, C., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2017; Volume 10284, pp. 166–175.
24. Gibaldi, A.; Vanegas, M.; Bex, P.J.; Maiello, G. Evaluation of the Tobii EyeX Eye tracking controller and Matlab toolkit for research. *Behav. Res. Methods* **2017**, *49*, 923–946. [CrossRef] [PubMed]
25. Blascheck, T.; Kurzhals, K.; Raschke, M.; Burch, M.; Weiskopf, D.; Ertl, T. State-of-the-art of visualization for eye tracking data. Proceedings of Eurographics Conference on Visualization (EuroVis), Swansea, UK, 9–13 June 2014; pp. 63–82.
26. Špakov, O.; Miniotas, D. Visualization of eye gaze data using heat maps. *Electron. Electr. Eng.* **2007**, *2*, 55–58.
27. Maurus, M.; Hammer, J.H.; Beyerer, J. Realistic heatmap visualization for interactive analysis of 3D gaze data. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14), 14–18 March 2014; ACM: New York, NY, USA, 2014; pp. 295–298.
28. Duchowski, A.T.; Price, M.M.; Meyer, M.; Orero, P. Aggregate gaze visualization with real-time heatmaps. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12), Santa Barbara, CA, USA, 28–30 March 2012; ACM: New York, NY, USA, 2012; pp. 13–20.
29. Tula, A.D.; Kurauchi, A.; Coutinho, F.; Morimoto, C. Heatmap explorer: An interactive gaze data visualization tool for the evaluation of computer interfaces. In Proceedings of the 15th Brazilian Symposium on Human Factors in Computer Systems, IHC'16, 4–7 October 2016; ACM: New York, NY, USA, 2016; Volume 24, pp. 1–9.
30. Kurzhals, K.; Hlawatsch, M.; Heimerl, F.; Burch, M.; Ertl, T.; Weiskopf, D. Gaze Stripes: Image-Based Visualization of Eye Tracking Data. *IEEE Trans. Vis. Comput. Gr.* **2016**, *22*, 1005–1014. [CrossRef] [PubMed]
31. Burch, M.; Kumar, A.; Mueller, K.; Weiskopf, D. Color bands: Visualizing dynamic eye movement patterns. In Proceedings of the 2016 IEEE Second Workshop on Eye Tracking and Visualization (ETVIS), Baltimore, MD, USA, 23 October 2016; pp. 40–44.
32. Kurzhals, K.; Hlawatsch, M.; Seeger, C.; Weiskopf, D. Visual Analytics for Mobile Eye Tracking. *IEEE Trans. Vis. Comput. Gr.* **2017**, *23*, 301–310. [CrossRef] [PubMed]
33. Anderson, N.C.; Anderson, F.; Kingstone, A.; Bischof, W.F. A comparison of scanpath comparison methods. *Behav. Res. Methods* **2014**, *47*, 1377–1392. [CrossRef] [PubMed]
34. Raschke, M.; Chen, X.; Ertl, T. Parallel scan-path visualization. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12), Santa Barbara, CA, USA, 28–30 March 2012; ACM: New York, NY, USA, 2012; pp. 165–168.
35. Pfeiffer, T.; Memili, C. Model-based real-time visualization of realistic three-dimensional heat maps for mobile eye tracking and eye tracking in virtual reality. In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16), Charleston, SC, USA, 14–17 March 2016; ACM: New York, NY, USA, 2016; pp. 95–102.
36. Voßkühler, A.; Nordmeier, V.; Kuchinke, L. OGAMA (Open Gaze and Mouse Analyzer): Open-source software designed to analyze eye and mouse movements in slideshow study designs. *Behav. Res. Methods* **2008**, *40*, 1150. [CrossRef] [PubMed]
37. Netzel, R.; Weiskopf, D. Hilbert attention maps for visualizing spatiotemporal gaze data. In Proceedings of the 2016 IEEE Second Workshop on Eye Tracking and Visualization (ETVIS), Baltimore, MD, USA, 23 October 2016; pp. 21–25.
38. Drusch, G.; Bastien, J.C.; Paris, S. Analysing eye-tracking data: From scanpaths and heatmaps to the dynamic visualisation of areas of interest. *Adv. Sci. Technol. Higher Educ. Soc. Concept. Age* **2014**, *20*, 205.

39. Choi, J.; Oh, T.; Kweon, I.S. Human attention estimation for natural images: An automatic gaze refinement approach. *arXiv*, **2016**, arXiv:1601.02852.
40. Quaia, C.; Optican, L.M. Three-dimensional rotations of the eye. In *Adler's Physiology of the Eye: Clinical Application*, 10th ed.; Kaufman, P.L., Alm, A., Eds.; Mosby: New York, NY, USA, 2002; pp. 818–829.
41. Schor, C.M.; Maxwell, J.S.; Stevenson, S.B. Isovergence surfaces: The conjugacy of vertical eye movements in tertiary positions of gaze. *Ophthalmic Physiol. Opt.* **1994**, *14*, 279–286. [CrossRef] [PubMed]
42. Haslwanter, T. Mathematics of three-dimensional eye rotations. *Vision Res.* **1995**, *35*, 1727–1739. [CrossRef]
43. Tobii, Accuracy and Precision, Test Report, Tobii T60 XL Eye Tracker. 2011. Available online: http://www.tobii.com/ (accessed on 8 August 2018).
44. Sheldon, M.R. *Introductory Statistics*, 4th ed.; Elsevier: Cambridge, MA, USA, 2017.
45. Imamura, K.; Kuroda, H.; Fujimura, M. Criterial image preparation for a search method for digital watermarking image using correlation coefficient between pixel value histograms. In Proceedings of the 2013 9th International Conference on Information, Communications & Signal Processing, Tainan, Taiwan, 10–13 December 2013; pp. 1–5.
46. Jiang, Q.-r.; Gao, Y. Face recognition based on Detail Histogram Intersection kernel. In Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems, Shanghai, China, 20–22 November 2009; pp. 71–74.
47. Mahalingam, T.; Mahalakshmi, M. Vision based moving object tracking through enhanced color image segmentation using Haar classifiers. In Proceedings of the Trendz in Information Sciences & Computing (TISC2010), Chennai, India, 17–19 December 2010; pp. 253–260.
48. Hansen, D.W.; San Agustin, J.; Villanueva, A. Homography normalization for robust gaze estimation in uncalibrated setups. In Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10), Austin, TX, USA, 22–24 March 2010; ACM: New York, NY, USA, 2010; pp. 13–20.
49. Mantiuk, R.; Kowalik, M.; Nowosielski, A.; Bazyluk, B. Do-It-Yourself Eye Tracker: Low-Cost Pupil-Based Eye Tracker for Computer Graphics Applications. *Adv. Multimed. Model.* **2012**, *7131*, 115–125.
50. Narcizo, F.B.; Hansen, D.W. Depth Compensation Model for Gaze Estimation in Sport Analysis. In Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 13–16 December 2015; pp. 788–795.
51. Metz, C.E. Basic principles of ROC analysis. In *Seminars in Nuclear Medicine*; WB Saunders: Philadelphia, PA, USA, 1978; Volume 8, pp. 283–298.
52. Turner, D.A. An intuitive approach to receiver operating chararcteristic curve analysis. *J. Nucl Med.* **1978**, *19*, 213–220. [PubMed]
53. Park, S.H.; Goo, J.M.; Jo, C.-H. Receiver Operating Characteristic (ROC) Curve: Practical Review for Radiologists. *Korean J. Radiol.* **2004**, *5*, 11–18. [CrossRef] [PubMed]
54. T Lasko, T.A.; Bhagwat, J.G.; Zou, K.H.; Ohno-Machado, L. The use of receiver operating characteristic curves in biomedical informatics. *J. Biomed. Inform.* **2005**, *38*, 404–415. [CrossRef] [PubMed]
55. Qin, Z. ROC analysis for predictions made by probabilistic classifiers. In Proceedings of the International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18–21 August 2005; Volume 5, pp. 3119–3124.
56. Blascheck, T.; Burch, M.; Meisel, T.; Schneider, T.; Mumin, S. Exploring Eye Movements with Node-Link Graph Layouts. In Proceedings of the Workshop on Eye Movements for Spatial Research (ET4S), Zurich, Switzerland, 14 January 2018.
57. Poole, A.; Ball, L.J. Eye Tracking in Human Computer Interaction and Usability Research: Current Status and Future Prospects. In *Encyclopedia of Human Computer Interaction*; Ghaoui, C., Ed.; Idea Group Reference: Hershey, PA, USA, 2005; pp. 211–219.
58. Liu, J.; Tang, T.; Wang, W.; Xu, B.; Kong, X.; Xia, F. A Survey of Scholarly Data Visualization. *IEEE Access* **2018**, *6*, 19205–19221. [CrossRef]
59. Qin, X.; Luo, Y.; Tang, N.; Li, G. DeepEye: An automatic big data visualization framework. *Big Data Min. Anal.* **2018**, *1*, 75–82. [CrossRef]
60. Butcher, P.W.S.; Ritsos, P.D. Building Immersive Data Visualizations for the Web. In Proceedings of the 2017 International Conference on Cyberworlds (CW), Chester, UK, 20–22 September 2017; pp. 142–145.

61. Barz, M.; Daiber, F.; Bulling, A. Prediction of gaze estimation error for error-aware gaze-based interfaces. In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications, Charleston, SC, USA, 14–17 March 2016; pp. 275–278.
62. Milovanović, I. *Python Data Visualization Cookbook*; Packt Publishing: Birmingham, UK, November 2013; 280p.
63. Keister, D.M.; Larson, D.; Dostal, J.; Baglia, J. The Radar Graph: The Development of an Educational Tool to Demonstrate Resident Competency. *J. Grad. Med. Educ.* **2012**, *4*, 220–226. [CrossRef] [PubMed]
64. Adams, C.R. *Learning Python Data Visualization*; Packt Publishing: Birmingham, UK, 2014.
65. Meier, B. *Python GUI Programming Cookbook*; Packt Publishing: Birmingham, UK, 2017.

Appendix E: GazeVisual – a Practical Software Tool and Web Application for Performance Evaluation of Eye Tracking Systems

IEEE Transactions on Consumer Electronics

GazeVisual – a Practical Software Tool and Web Application for Performance Evaluation of Eye Tracking Systems

Journal:	*Transactions on Consumer Electronics*
Manuscript ID	TCE-2019-03-0092
Manuscript Type:	Original Article
Date Submitted by the Author:	07-Mar-2019
Complete List of Authors:	Kar, Anuradha; National University of Ireland Galway, College of Engineering and Informatics; National University of Ireland, Galway Corcoran, Peter; National University of Ireland Galway College of Science, Engineering & Informatics;
Keywords:	eye tracking, performance evaluation, data quality, cloud computing, visualizations

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

GazeVisual – a Practical Software Tool and Web Application for Performance Evaluation of Eye Tracking Systems

Anuradha Kar, *Student Member, IEEE*, and Peter Corcoran, *Fellow, IEEE*

Abstract— **The concept and functionalities of a software tool developed for in depth performance evaluation of eye gaze estimation systems is presented. The software, *GazeVisual* has capabilities for quantitative, statistical and visual analysis of eye gaze data as well as generation of static and dynamic visual stimuli for sample gaze data collection. This is a first of its kind cross-platform tool for gaze data analysis & evaluation. This software is made freely available to the eye gaze research and development community to provide a common framework for estimating the quality and reliability of data from eye tracking systems, especially those implemented in consumer electronics (CE) applications. The feasibility of using this software is tested through case studies which show that the software can handle eye gaze datasets obtained from several different consumer grade eye trackers. GazeVisual operates consistently, irrespective of the platform, algorithm or hardware of the eye trackers. In addition, the GazeVisual software capabilities are also made accessible via a web based application enabling performance evaluation of eye tracker data over a cloud-based platform.**

***Index Terms*— Eye tracking, eye gaze, performance evaluation, data quality, visualizations, web-application**

I. INTRODUCTION

EYE trackers working on various consumer platforms including desktop, automotive, wearable and handheld devices are frequently affected by factors like user head movements, platform orientations, user distance and display properties.[1]–[3]. Currently there exist no comprehensive tools or software that may be used to evaluate an eye tracker's data quality, or quantitatively estimate the impacts of practical operating conditions on its performance. This makes it nearly impossible for eye gaze researchers, engineers and application developers to understand and compare the capabilities and limits of various eye trackers, know their robustness and reliability or study their gaze data quality [4].

This work presents a newly developed software tool, named *GazeVisual* that incorporates a suite of statistical and visualization functions for detailed evaluation of consumer grade eye trackers. It is a desktop software application having a graphical user interface (GUI) and software components that can be used to input and process files containing data from commercial eye trackers and inform a user about the accuracy and limits of their systems. It can generate visual stimuli to collect sample data from an eye tracker and has possibilities for direct interfacing with an eye tracker for data logging and evaluation. The GazeVisual software only requires a sample of gaze and ground truth data from an eye tracker to produce multiple numerical and graphical data evaluation results. All parts of the software are written in Python language and the full source code is provided in an open repository for gaze researchers and engineers to use and upgrade.

In addition to the desktop GUI, a web-application named GazeVisualApp is also developed to implement a basic set of functions present in the GazeVisual software over a cloud based Python environment. This has several advantages, which are explained in more details later in this paper; e.g. eye tracker evaluation independent of operating system (OS) or platform and ubiquitous accessibility.

Early-stage development of this software was reported in our recent works [5][6], and this paper builds on the same concepts with additional and improved features. Sample data, source code and demo videos of the GazeVisual GUI software may be found in the GitHub repository with the address: https://github.com/anuradhakar49/GazeVisual-Lib.

II. PRIOR ART IN GAZE DATA ANALYSIS SOFTWARE

Eye tracking has found uses in a variety of consumer electronics (CE) applications such as in automotive (for driver monitoring) [7], augmented and virtual reality [8] (for foveated rendering and immersive experiences), smartphones and TV [9][10] (for gaze based password entry, menu selection and navigation). Recent works in this field include [11] where gaze patterns are used to assess how easily users can navigate an interface of a connected self-injection system. Gaze data is used as an indicator to study the usability, efficiency and ease of use of the drug delivery device. Similarly, the design effectiveness of observation charts in a hospital is evaluated in [12] by comparing viewing patterns of users derived from eye tracking data. Eye movements and pupillary response are used as indicators of cognitive load while users answered mathematical questions in [13] and for studying cognitive processes and learning aspects in [14]. In [15], the influence of emotions on the visual acuity of users was studied which showed that eye movements like fixations and saccades clearly respond to levels of stress.

Manuscript submitted on 6th March for review. This research work is funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded by FotoNation Ltd. Project ID: 13/SPP/I2868 on "*Next Generation Imaging for Smartphone and Embedded Platforms*".

A. Kar is with the Department of Electrical and Electronic Engineering, National University of Ireland, Galway, H91TK33, Ireland. (e-mail: a.kar2@nuigalway.ie)

P. Corcoran is with the Department of Electrical and Electronic Engineering, National University of Ireland, Galway, H91TK33, Ireland. (e-mail: peter.corcoran@nuigalway.ie)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

It was discussed in [4] that eye trackers, especially in CE applications face a number of challenging conditions, such as head pose variations, platform movements, user distance variations, variable display properties to name a few. Under such unconstrained operations, gaze estimation errors become large and also the level of noise and outliers in gaze data increase. This poses serious difficulties towards the reliable design of gaze applications and hampers the prospect of using eye gaze in CE use cases. Successful use of eye trackers or gaze data in CE applications therefore requires frequent and in-depth measurement of gaze data quality and studying its variability under different operating conditions.

However, the problem is that, at present there exist no dedicated tools or software that can be used to readily evaluate an eye tracker's accuracy levels and data quality. A survey of several commercial and open-source software tools for gaze data analysis is presented in Table I. The survey reveals that most data analysis software for eye trackers are built for exploration of eye movement characteristics and cognitive processes, but not towards objective evaluation of gaze data quality. This forms the motivation for developing the GazeVisual software, whose design and functionalities are described henceforth.

TABLE I
A SURVEY OF SOFTWARE TOOLS FOR EYE GAZE DATA ANALYSIS

Ref.	Name & type of software tool	Description of main features and capabilities of the software tool
[16]	iComponent Type: Free application	Recording and analysis of data from multiple eye trackers. Four types of 2D and one 1D visualization of gaze data possible. Heatmap, clustering and gaze path replay allowed.
[17]	Heatmap explorer Type: Open source	Study user gaze behavior, visual attention and focus. Produces video with heatmap, display user visual focus on various backgrounds
[18]	Ogama Type: Open source	Recording and analyzing of eye and mouse tracking data, preprocessing and filtering, creation of attention maps, areas-of-interest.
[19]	Pygaze Type: Open source	Creating eye tracking experiments, presenting visual and auditory stimuli, response collection via keyboard, mouse, joystick, saccade detection.
[20]	GazeParser Type: Open source	Gaze position and saccade detection library using webcams.
[21]	GazeAlyze, Type: Open source	MATLAB toolbox for the analysis of eye movement data, detecting events, filtering of artefacts, generating regions of interest, path plots and fixation heat maps.
[22]	Eyemap Type: Free application	Processing binocular eye movement data, filtering, and processing gaze data from reading and psychology experiments, multiple eye tracker support.
[23]	Software by eye tracker company-1 Type: Commercial	Stimuli presentation, creating eye tracking experiments, replay eye tracking videos, overlay with gaze points, adjustable fixation filters, create AOI statistics, tables, charts.
[24]	Software by eye tracker company-2 Type: Commercial	Create heatmap and fixation maps, display AOI regions, viewed time, revisits, create AOI statistics, web data analysis, dynamic video creation.
[25]	Custom gaze data analysis software-2 Type: Commercial	Create stimuli, capture eye data, mouse clicks, scroll, key presses, estimate fixations, blinks, cognitive workload, pupil size.

III. CONCEPT AND DESIGN OF THE GAZEVISUAL SOFTWARE

A. Rationale for development of the GazeVisual software

The main objectives of developing the GazeVisual software are: 1. Providing methods for detailed analysis and visualizations of gaze data quality, accuracy and gaze error patterns 2. Evaluating multiple gaze datasets from one or more trackers or users 3. Implementing gaze data evaluation methods irrespective of eye tracking algorithm, hardware or application by just using gaze and ground truth data samples 4. Generation of static or dynamic stimuli for live gaze data capture and possibility of direct collection of sample data from an eye tracker under test 5.Integration of all these components via an easy to use GUI, which allows creation and saving of plots and results. 6. Providing an open-source software for eye tracker evaluation to gaze researchers and developers.

B. GazeVisual concept, architecture and dataflow

For data collection from an eye tracker, a user is positioned in front of the tracker which is mounted on the display of a computer (Fig. 1a). During a testing session, a set of visual targets are presented on the display and the eye tracker under test continuously records the gaze data of the user as they look at these targets (or stimuli). The recorded user gaze coordinates form the test data and the on-screen stimuli coordinates form the ground truth data. These two datasets are the key inputs used by GazeVisual to implement various metrics and visualizations and produce evaluation results. Conceptually, GazeVisual software has 4 modules (Fig.1b):

1. A test user interface (test UI) which displays visual targets on a screen where user gaze is tracked.
2. A data collection module (DCM) which logs data from an eye tracker under test while the test UI runs.
3. A data processing module (DPM) which implements numerical metrics and visualizations on collected data
4. The tool GUI via which a tester can upload data, access evaluation tools, display and save results/ plots.

The dataflow hierarchy model within the GazeVisual tool is shown as the pyramid in Fig. 1c. The lowest level has the raw gaze and ground truth data and relevant system variables, from which the gaze error values are computed. Final output level comprises of various accuracy metrics and visuals that serve as performance indicators of the eye tracker under test.

C. Software dependencies, hardware requirements

All components of the GazeVisual GUI tool are written in Python language and require Python libraries like Numpy, Matplotlib, Tkinter, Pygame Statsmodels, Sklearn and Seaborn. It can run as a desktop application on any operating system having Python 2.7. For eye tracker data collection, the tracker's Software Development Kit (SDK) and calibration routines must be installed in the user computer. Else GazeVisual can use gaze data which has been collected from the eye tracker beforehand. GazeVisual has been tested to run on ordinary PCs with 3.6 GHz processor, with 16 GB RAM. To use the live tracking feature, it needs a suitable USB (2.0 or 3.0) port for connecting the eye tracker and a Python based SDK of the tracker to communicate with.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

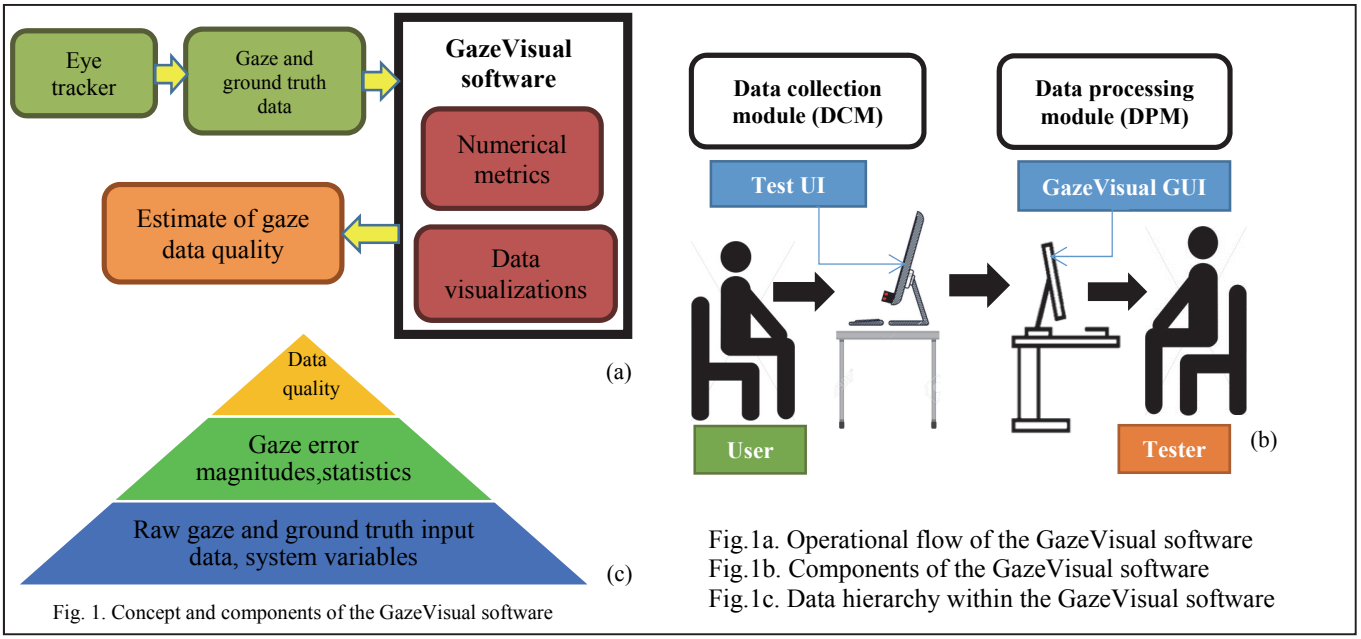


Fig. 1. Concept and components of the GazeVisual software

Fig.1a. Operational flow of the GazeVisual software
Fig.1b. Components of the GazeVisual software
Fig.1c. Data hierarchy within the GazeVisual software

D. Input data format

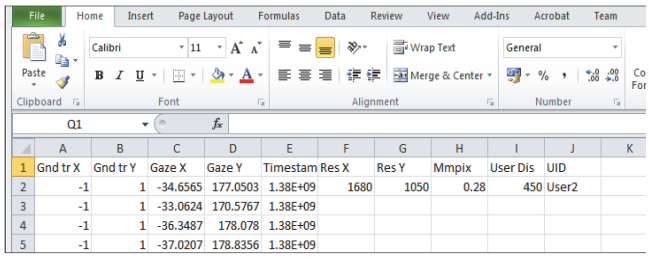
The input data file format for the GazeVisual software is CSV (comma separated values) The first two columns of the file must contain the x, y pixel values of ground truth data, followed by two columns of gaze x, y pixel coordinates, followed by a column having the data collection timestamps for each data point and then two columns having the X, Y resolution of the display area used for collecting the data (ResX and ResY for X,Y screen dimensions in Fig.2). Next a column having screen pixel size (μ), a column specifying user-tracker distance (in mm), and finally a column with a data identifier, e.g. user /experiment name have to be included. This input file format is followed for implementation of both the GUI as well as the web application. Care must be taken that the data input file must not have blank entries, the display center is chosen as the data origin and header fields for each data column are kept as shown below in Fig 2. Several sample data files saved in this format are in the GitHub repository's "Sample data" folder, whose link is provided above.

The "mmpix" or millimeter per pixel (μ) value of the used computer display may be computed using Eqn. 1a and 1b, where w_p and h_p are the screen width and height in pixels:

$$\mu = d_m / d_p \quad (1a)$$

$$d_p = \sqrt{w_p^2 + h_p^2} \quad (1b)$$

where dm is the screen diagonal size in millimeters



	A	B	C	D	E	F	G	H	I	J
1	Gnd tr X	Gnd tr Y	Gaze X	Gaze Y	Timestam	Res X	Res Y	Mmpix	User Dis	UID
2	-1	1	-34.6565	177.0503	1.38E+09	1680	1050	0.28	450	User2
3	-1	1	-33.0624	170.5767	1.38E+09					
4	-1	1	-36.3487	178.078	1.38E+09					
5	-1	1	-37.0207	178.8356	1.38E+09					

Fig.2. Sample input data format accepted by GazeVisual

IV. Organization, Layout and Functionalities of the GazeVisual Software

All functionalities of GazeVisual are organized under four windows (Fig.3). The major analytical and graphical functions are in the "Data analysis" and "Visualizations" windows. The Test UI window contains functions for generating static or dynamic stimuli, connecting to an eye tracker and saving of the collected gaze data. Each window works independently and is easy to navigate and use by a generic user.

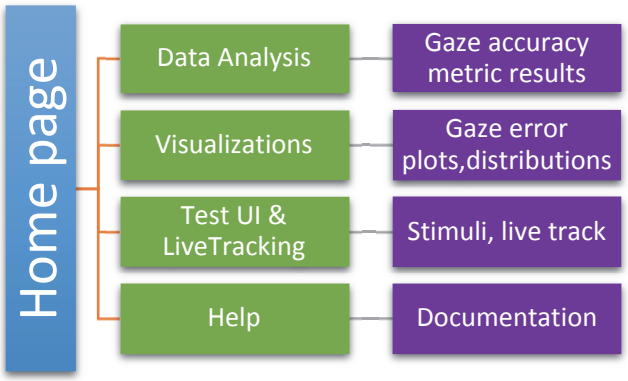


Fig.3. Four windows of GazeVisual GUI tool and their functions.

The "Home page" provides a brief description of the software's capabilities. The "Help" window provides information on data input, output, formatting, references and link to the GitHub repository for this project. Descriptions of the other windows and their components are presented below. Demo videos showing the functions of the different windows of the tool may be found in the GazeVisual GitHub repository.

A. Data Analysis window

This window provides functions that allow users to upload gaze data CSV files (in the format described in Section III.D) to the software and estimate gaze tracking accuracy and quality from the uploaded dataset. Several gaze data analysis methods are provided, the results of which may be viewed in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

the "Output Console" and plot area of the window. Using the input gaze and ground truth data, gaze angle (at each data point) and gaze error statistics like mean, standard deviation, Z-score, 95% confidence interval and values of gaze yaw/pitch angular variables are estimated using the formulae as below:

$$\text{Gaze.Angle } (\theta) = \tan^{-1}(\text{OSD}/Z) \quad (2)$$

$$\text{Gaze pitch angle } = \tan^{-1}(\text{GazeY}/Z) \quad (3)$$

$$\text{Gaze yaw angle} = \tan^{-1}(\text{GazeX}/Z) \quad (4)$$

Where GazeX, GazeY are the gaze point coordinates, Z is the user distance from the screen-tracker setup OSD is the on-screen distance of a user's gaze point from the screen origin. The gaze error at each data point, which forms the main parameter for further analysis in the software is given by:

$$\text{Gaze error } (x_i) = \text{Gaze.Angle} - \text{GT. Angle} \quad (5)$$

where GT.Angle is the angular position of each ground truth data point. The statistical operations made on gaze data are

$$\text{Mean error}(\varphi): \ 1/n \sum_{i=1}^{n} x_i \quad (6)$$

$$\text{Z-score } (\sigma): \ (x_i - \varphi)/\sigma \quad (7)$$

$$95\% \text{ confidence interval: } \ \varphi \pm 1.96\sigma/\sqrt{n} \quad (8)$$

$$\text{KDE} = \sum_{i=1}^{N} K((y - x_i)/h \quad (9)$$

Where K is a Gaussian Kernel defined as:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-0.5u^2} \quad (10)$$

σ is the error standard deviation and n is the number of data points. KDE is the Kernel Density Estimate computed using a Gaussian Kernel (K) and bandwidth (h) of 0.2 [20] on gaze error values. Detailed explanations on gaze data analysis methods may be found in our previous work [6], Section 4. A view of the "Data Analysis" window is shown in Fig.4 and meanings of text labels and buttons on it are described below:

a. Load gaze data: Beside this text label is the "Upload csv file" button which allows users to browse and select the gaze data input CSV file and upload its column values to the GUI tool. A scatter plot of the gaze data overlaid with ground truth data is then automatically created in the plot area.

b. Analysis results/plots: There are two buttons beside this text label. After uploading the gaze data file, the analysis results can be obtained by clicking the "Statistics" button, with which users will be shown a plot of the estimated gaze angular errors versus time (using timestamps in the input CSV file). Statistical measures estimated using Equations (6-8) will be displayed in the output console. With the "Yaw/Pitch" button, gaze yaw and pitch angles (Equations 3, 4), will be plotted and their statistics will be shown on the output console (Fig 4).

c. Gaze angles/ Z-score: The "Z score plot" button creates two subplots, one with gaze angles vs ground truth and the other having the Z-score for each data point.

d. Upload multiple files: The software allows uploading of two CSV data files for their comparison. Two different gaze datasets saved in two CSV files may be uploaded by clicking buttons "Data-1" and "Data-2", which will create scatter plots of gaze vs ground truth data for each dataset.

e. Compare datasets: Beside this label, clicking the "Statistics table" button produces a table displaying the data statistics from each dataset for direct comparison.

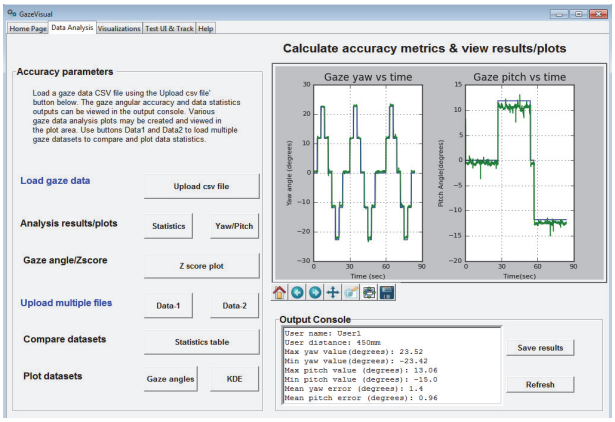


Fig.4. "Data Analysis" window showing the plot of gaze yaw and pitch angles and the output console area showing yaw/pitch statistical results.

f. Plot datasets: The "Gaze Angles" buttons next to this label displays gaze angles vs ground truth for the two gaze datasets. The "KDE" button estimates the 1D-KDE (Eqns. 9-10) on the gaze error from each dataset and plots them adjacently for comparison (e.g. shown in Fig. 8b below).

g. *"Output Console"* area: The various results of numerical analysis functions of the window are displayed in this area.

h. "Save results" button: Contents in the "Output Console" area may be saved in a text file by clicking this button.

i. "Refresh" button: This clears the "Output Console" and plot area of their current contents.

j. Plot area: This display area is present in both the Data Analysis and the Visualizations windows and is used to show output plots from the software functions. The plot area shows the x, y coordinates of plotted data points on hovering the cursor over the plots/ subplots. Buttons embedded below the plot area allow users to pan and zoom into plot areas using the "Pan Axes", "Zoom to Rectangle" buttons. All plots can be saved in PNG file formats using the "Save the Figure" button.

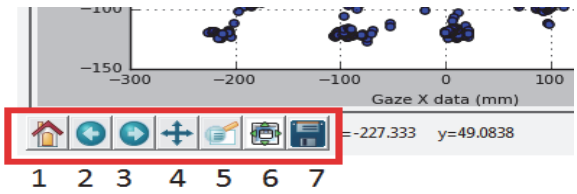


Fig.5. Buttons in the plot area of the tool. Buttons 2 and 3 are for moving between plot views and 6 is for formatting subplots. 1,4,5,7 are for resetting original view, panning axes, zooming and saving the plot.

B. Visualizations window

Data aggregation and graphical presentation are essential for understanding the characteristics of gaze data that are collected in large volumes during experiments. GazeVisual software has several visualization functions for studying gaze data quality in the "Visualizations" window. Functions of the labels and buttons in this window (Fig. 6) are explained here:

*a. Load gaze data: T*he "Upload csv file" button allows users to upload gaze data CSV file values to the software.

b. Error histogram (and text box alongside): The "Plot" button beside this label plots the histogram of gaze error values (using the uploaded data) with the bin size value entered in the box alongside. The default bin size is 20.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

c. Data density plot, Mean/SD, 3D plot: Using the "Data density plot" button will plot the uploaded raw gaze data as data point clusters, color-mapped according to point densities. A color bar would show the number of points in a cluster mapped by color. The "Mean/SD" button creates an error bar plot, which shows the magnitude and standard deviation of gaze error by sampling every 100 data points. The 3D plot button creates a plot with the magnitudes of gaze errors (in degrees) plotted along the Z axis as a function of X and Y dimensions (in pixels) of the display screen (Fig. 6). These plots help to show the local error patterns in the gaze dataset, visualize data quality/noise and detect any irregularity.

d. Upload multiple files: The Visualizations window also allows uploading of two different gaze datasets for which the "Data-1" and "Data-2" buttons are provided.

e. Compare datasets: Beside this label, the button "Histograms" plots the gaze error histograms of the two uploaded datasets using the bin size entered in the text box above. The "Regression" button creates a regression plot of the gaze errors from the two datasets. The "Bar plots" button may be used to plot the mean gaze error, minimum and maximum gaze angles for the two datasets as vertical bars, side by side. The "Box plots" button may be used to display and compare several gaze error statistical attributes (e.g. minimum, first quartile, median, third quartile, and maximum) by creating box-plots from the two datasets [6]. The "Error line plot" button shows gaze errors from the two datasets for each data point. This window also provides several options for changing plot color, background, addition of text annotations and setting of color maps. These functions are found under "Alter plot features". The "Refresh" button and ones below plot area has similar functions as in the Data Analysis page.

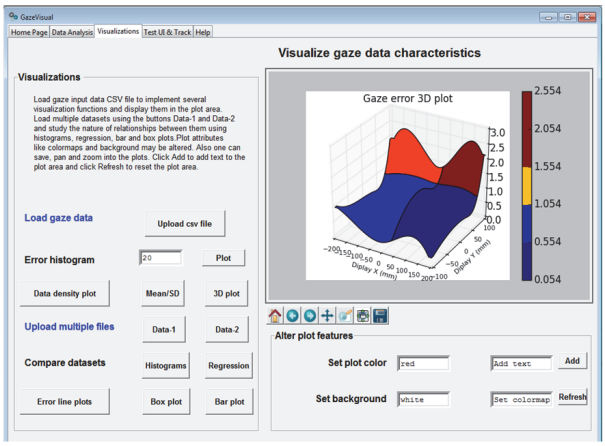


Fig. 6. "Visualizations" page of GazeVisual showing a 3D gaze error plot.

In gaze based consumer applications, poor gaze data quality may result from large gaze estimation errors, data loss or scatter, tracker noise etc. These factors may cause failure or reliability issues in the overall gaze application. To ensure the quality of gaze data at all times, the evaluation functions of the Data Analysis and Visualization windows of the GazeVisual software could be beneficial. Using them, engineers and system developers can easily test the noise, scatter or accuracy levels in their gaze data, before and while using the data in their applications and also detect sources of the noise/errors.

C. Test UI and LiveTracking window

The "Test UI and LiveTracking" window incorporates two functions: the first one is for generating visual stimulus for sample gaze data collection from an eye tracker (Test UI section) and the second one is to allow direct interfacing of an eye tracker with the GazeVisual software for direct data-logging (LiveTracking section).

In contemporary gaze research, there are no standard or common stimuli that are used for gaze data collection and researchers build their own stimuli routines to test their eye tracking device or algorithm. This leads to difficulties in comparing datasets obtained from different researchers due to differences in UI characteristics, e.g. dissimilar stimuli target size, patterns, velocity or visual eccentricities, which may strongly affect eye tracking accuracy [6]. This forms the motivation for incorporating the Test UI section in GazeVisual, in order to provide a common platform for stimuli generation for gaze data collection from eye trackers.

In the Test UI section, it is possible to configure and generate two static UIs and a dynamic UI, which comprise of visual stimuli or targets either fixed or moving on different locations of the display screen. The on-screen locations of the UI targets are known and form the ground truth (x, y) coordinates, which can be saved in CSV files.

The first two text boxes in the section allow entering of the size of the UI window in pixels for X,Y dimensions. Then there are options for setting the size of the stimulus targets in pixels and setting the background color of the stimuli window.

The Static UI-1 comprises of clickable buttons on the display screen where a user has to look and click. The Static UI-2 comprises of a ball appearing discretely at fixed locations on the Test UI window at certain time intervals. This time interval (in seconds) may be changed by putting the desired value in the 2nd test box beside the label saying "Set stimulus size/speed". The Dynamic UI comprises of a ball moving continuously across the window in multiple rows. The purpose of having the different UIs is so that both discrete (fixations) and smooth eye movement data may be collected for tracker evaluations. Demonstrations of the Static and Dynamic UIs with the preset attributes may be made on the adjacent display area of the interface by pressing the "Demo" buttons. On pressing the "Start" button for one of the UIs, a separate window is created to display the target stimuli to users as in Fig. 7a (here Test UI window size is 400x 400 pixels).

The LiveTracking section as shown in Fig.7b presents a new concept of direct eye tracker data collection and evaluation. This section allows direct interfacing of an eye tracker with the GazeVisual software to collect gaze data and instantly evaluate the data characteristics. The LiveTracking feature comprises of a data-logging routine for eye trackers synchronized with one of the Test UIs described above. Currently this is possible only for an eye tracker whose SDK is available in Python language or which have a Python API for communication with the tracker device. For trackers which do not have a Python SDK, users may synchronize their tracker with a Test UI for gaze data collection.

As a proof of concept of the Live Tracking feature, a remote eye tracker which has Python library support for data logging is interfaced with the GazeVisual software and an example

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

operation of this function with Static UI-1 is in Fig.7b(i). After connecting the eye tracker and positioning a user in front of it (Fig. 1), the "Start Tracking" button is clicked. With this, the Static UI-1 is launched and user gaze data is collected from the eye tracker as the user clicks the UI buttons.

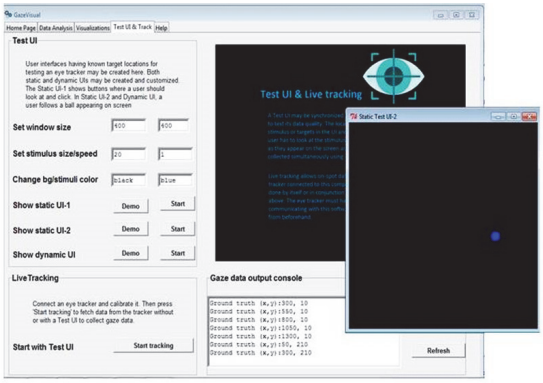
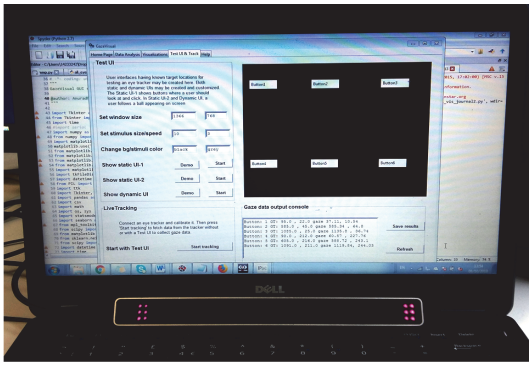


Fig.7a. TestUI section of the GazeVisual software showing an instance of the Static UI-2 in a small 400x400 pixel window with the blue stimulus ball.

Gaze data from the tracker and click locations can be seen in the "Gaze data output" console of Fig. 7b(ii). "GT" refers to the (x, y) coordinates of button click positions and values after "Gaze" are gaze data recorded by the eye tracker. A demo video of this function may be found in the GitHub repository.



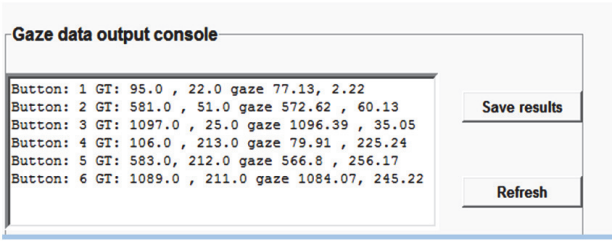


Fig.7b(i). Shows an eye tracker connected to the computer to test the LiveTracking feature and a demo of the Static UI-1 in the "Test UI and LiveTracking" window of the GazeVisual software. 7b(ii) shows the "Gaze data output console" displaying the ground truth and corresponding gaze locations tracked by the connected eye tracker after running Static UI-1.

D. Discussions

Eye tracking in consumer platforms like automotive, handheld devices and AR/VR faces a lot of challenges during practical operations owing to factors like user distance, physical movement, head pose variations and platform orientation or vibrations. These factors may have large and unpredictable impacts on gaze accuracy and gaze data quality. Therefore, gaze data needs to be constantly evaluated if eye tracking in these systems is to be done reliably. The GazeVisual software incorporates several new concepts which may be useful to researchers, engineers and generic users of consumer grade eye trackers for in-depth, continuous and fast evaluation of their eye tracker's data quality. The data analysis and visualization functions of this software may be used irrespective of eye tracking hardware or algorithm and the tool interface is also simple to use. It can also be used for setting up eye tracking experiments to collect sample gaze data for quality analysis. GazeVisual is a new kind of composite software, designed solely for the performance evaluation of consumer grade eye trackers using their data outputs. It allows understanding of an eye tracker's data quality and tracking performance without requiring users to do programming, manual scanning of data or invading the eye tracking device.

V. TESTING THE GAZEVISUAL SOFTWARE WITH CONSUMER GRADE EYE TRACKERS

In this section, the results from testing the GazeVisual software on gaze data from three different consumer grade eye trackers are presented as case studies below. The tests are done to ensure that the software can handle gaze data from different eye tracker types and produce consistent numerical results and visualizations. The tests also demonstrate how the GazeVisual software can quantitatively show the difference in quality of gaze data from the different consumer eye trackers, operating on different platforms, e.g. desktop, tablet and head-mounted setups. For the tests, eye gaze datasets are collected from 20 participants using two remote eye trackers and a head mounted tracker following the protocol shown in Fig. 1. All data were saved in the format described in Section III.D. Results from these studies are in Table II.

A. Case study 1: Evaluating data from a remote eye tracker

Data from a commercial remote eye tracker (R1, such as described in [21]) with a frame rate of 30 Hz is collected from a user seated at 45 cm from a desktop screen showing gaze targets (Fig. 8a). The tracker is mounted on the screen (size: 22inch, resolution: 1680x 1050 pixels) and the collected data is used with GazeVisual. Fig. 4 shows Yaw and Pitch angles estimated from this data and displayed on the Data Analysis window of the GazeVisual software.

B. Case study 2: Comparing eye tracking data from two remote eye trackers

In this, data from two commercial remote eye trackers (R1, described in [21], R2 in [22]) are input to GazeVisual. For data collection, the two eye trackers were mounted on a desktop screen (22inch size, 1680x 1050 pixels resolution) during two separate sessions. In both cases, a participant is seated at 60 cm from the tracker (Fig. 8a). Fig. 8b shows the comparison of error distributions of gaze data obtained from the two trackers using "KDE" feature of the software (Eqns. 9, 10). It is seen that magnitudes and distribution of gaze errors from tracker R2 are quite different from that of R1 (Table II).

C. Case study 3: Comparing eye tracking data obtained from a desktop and a tablet platform using the same eye tracker

In this, a single remote eye tracker (R1) is mounted first on a desktop and then on a tablet (display: 10inch, 1200x 800

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

pixels) respectively in two separate experiments (Fig. 8a). Gaze data is collected from users sitting at a distance of 60 cm from the tracker setup in each case. A comparison of gaze data characteristics from the two platforms is done using the "Box plots" feature of the Visualizations window (Fig. 8c). It is seen that error levels in the tablet-tracker setup are lower than the desktop for the same user distance and the inter-quartile range of gaze error for the tablet is also lower than the desktop setup (Table II). Also the tablet setup has a narrower visual angle compared to the desktop for the same user distance. This indicates that gaze data quality from the tablet setup is superior to that from the desktop setup.

D. Case study 4: Evaluating and comparing data from a binocular head-mounted (HM) eye tracker

Eye tracking data from a binocular head mounted eye tracker [23] is collected and evaluated using the GazeVisual software. Data is collected from the head-mounted tracker with a frame rate of 120Hz and eye camera resolutions of 640 x480 pixels, when a user is seated at a distance of 60 cm from the gaze targets (Fig.8a). The collected data is brought to the common format of Section III D, using Viewport transform [24], since the head-mounted tracker produces gaze output data in the form of Normalized Device Coordinates (NDC). The NDC (Xndc, Yndc) is converted to centered screen pixel coordinates (Xp, Yp) using the following relations:

Xp = (Xndc + 1.0) * ScreenWidth * 0.5 + ScreenTopLeftX

Yp = (Yndc + 1.0) * ScreenHeight * 0.5 + ScreenTopLeftY

where ScreenTopLeftX and ScreenTopLeftY are (0,0). Fig 8d shows a comparison table (on Data Analysis page) using data from remote tracker R1(Dataset1) and HM tracker (Dataset2).

Fig.8a. Data collection from a desktop and a tablet (top) and a head mounted eye tracker (bottom). Chin rest is used for all cases. The blue screen with the black dots is the UI used to collect gaze data.

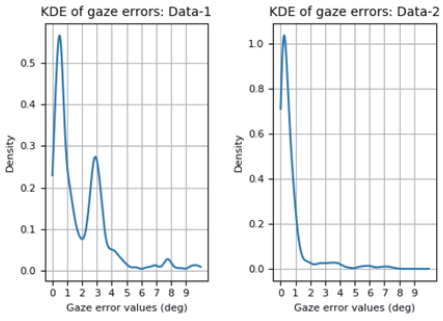


Fig.8b. KDE plots of gaze errors computed on data from two remote eye trackers for comparison using the "Data Analysis" window.

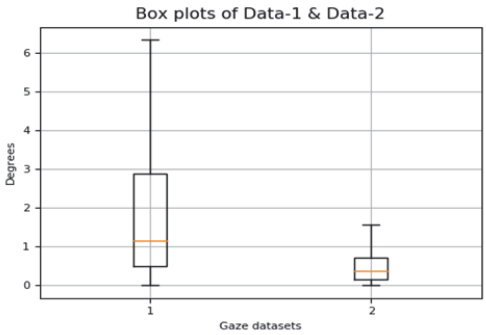


Fig.8c. Box plots created on the "Visualizations" window to compare gaze data collected from desktop (left) and tablet (right box) platforms.

TABLE II
Evaluation results from GazeVisual (columns 2, 4-6 in degrees)

Case study	Ac	Sd	Conf int	Max angle	Yaw (max,min)	Pitch (max-min)
R1+desktop	2.1	2.0	1.9-2.1	34.2	24, -26	12, -16
R2+desktop	7.3	6.0	8.8-9.1	33.0	22, -34	15, -24
R1+Tablet	1.2	0.8	1.1-1.2	13.6	9.23, -9.91	4.5, -6.5
HM tracker	4.4	0.9	4.4-4.5	41.3	33.76-25.04	24.7-20.4

E. Discussions

It is seen that GazeVisual can handle gaze data from different commercial eye trackers, provided their gaze data is organized in the format described in Section III.D. GazeVisual can quantitatively estimate and differentiate between the data quality of multiple trackers which could be highly useful for designers of consumer eye tracking applications to select the correct eye tracking device for their purposes, and also know about the practical limits and capabilities of their systems during un-constrained operations. Several data files used here are in the "Sample data" folder of the GitHub repository to let users understand the input data formatting. In Table II, Ac refers to angular accuracy and Sd is standard deviation.

VI. A WEB-APPLICATION FOR THE GAZE VISUAL SOFTWARE

To enhance the usability of the GazeVisual software, a web-application named GazeVisualApp is developed and deployed on a Python based cloud-server. This web-application has similar analysis and visualization capabilities as the GazeVisual desktop GUI application and accepts raw gaze data in the same CSV format to provide gaze data analysis results and visualizations through a web-browser. The concept and implementation of the web-application is described here.

A. Rationale for building a web-application for performance evaluation of eye trackers.

There are several benefits of having the capabilities of GazeVisual GUI tool available via a web-application (WA) These include: 1) Installation: Running the GazeVisual GUI software requires installation of Python 2.7 and several Python libraries. Many users may not have Python 2.7 or these libraries installed, or may not be familiar with Python language. Such users may be benefitted from the WA which would require only a web browser and internet connection to run. 2) Portability: Consumer platforms like tablets use mobile operating systems (OS) and may not have adequate hardware

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2912802, IEEE Transactions on Consumer Electronics

resources (processor speed, sufficient disk space and RAM) to and run Python packages. With the WA, the benefits of the GazeVisual software would be available to users irrespective of OS or device hardware. This could be especially handy for evaluating eye trackers operating on consumer handheld devices like smartphones and tablets which are highly dynamic platforms and gaze data quality on these platforms may get strongly affected by user motion, hand poses and head poses. 3)Flexible access, updates and support: With GazeVisualApp, the evaluation methods would be accessible to any user all the time and any updates will be immediately available to the users. 4)Continuous development: With the WA, eye tracker evaluation methods will be open to viewing and testing by the eye gaze community who may be able to contribute a stream of useful upgrade suggestions to include and test with the WA 5) Simplification: GazeVisualApp interface is simpler than its desktop version (although the later has more functionalities). Thus the WA could be more attractive to users who need only basic evaluation methods.

B. Concept and implementation of GazeVisualApp: A web-application for performance evaluation of eye trackers

The web-interface of GazeVisualApp is shown in Fig. 9a and using it is very simple. The interface comprises of instruction steps for using the application and a single file upload button bar. A user only needs to browse and upload a single CSV file having the gaze and ground truth data saved in the format described in Section III.D, via the file upload button. After uploading the CSV file, contents of the file are used to estimate gaze accuracy metrics and visualizations in an identical manner as done in the GazeVisual desktop application, using the same functions, algorithms and Python libraries. After uploading the CSV file, contents of the file are immediately displayed on the browser window, along with display of gaze data statistics and creation of six plots below the upload bar (Fig. 9b) The plots are: 1. gaze data vs ground truth scatter 2. gaze angles vs ground truth 3. gaze error vs time 4. gaze error distribution 5. gaze yaw angles and 6. gaze pitch angles vs respective ground truths. All functions of the web-application take place on a single web browser page. The gaze data used by the app is not stored in any database, but resultant plots may be saved in PNG formats during runtime. By refreshing the browser tab, the web-app may be restarted.

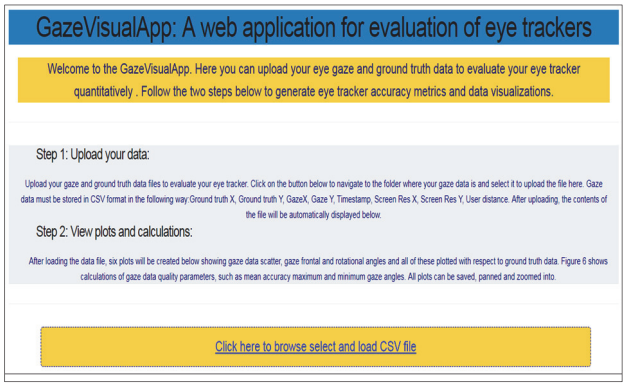


Fig.9a. The GazeVisualApp interface with the data upload bar.

GazeVisualApp is developed using Python 2.7 and Dash [25] which is an open source Python library (with MIT license) for creating interactive and analytical web applications. Dash is built on JavaScript libraries for building graphs and user interfaces, and Flask [26], which is a Python framework for creating web applications. Dash applications comprise of web servers running Flask and communicating JSON packets over HTTP requests. Dash allows use of interactive UI elements like buttons and graphs as well as coupling of Python data analysis libraries like Pandas, Numpy and Scipy. For building the GazeVisualApp, the frontend was built using UI elements provided by Dash (with HTML/CSS styles) and the backend comprised of various Python based analytical and data visualization components. Dash was suitable for building GazeVisualApp since the GazeVisual software functionalities could be easily ported to it, and the outputs of both the desktop and web applications could be made homogeneous and identical while working on the same gaze data files. Another benefit of using the Dash-Python framework is the good support of developers and resources, which may in future help to expand the GazeVisualApp to include more functionalities.

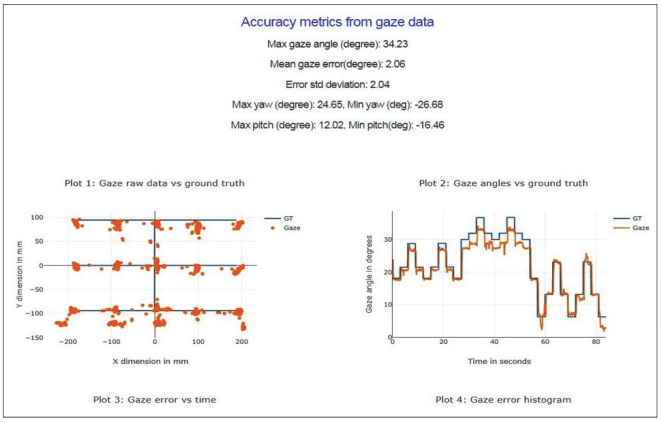


Fig.9b. Output plots and data statistics obtained from the GazeVisualApp after uploading gaze and ground truth data

C. *Deployment of the GazeVisualApp on a cloud platform*

GazeVisualApp is deployed on a cloud server to make it accessible to general users over internet. For this, a web hosting service provider (with a Platform as a service or PaaS model [27]) supporting Python language is chosen that allows apps built with Python and Flask (or any WSGI compliant framework) to run on hosted servers, and become available to multiple users. In the PaaS cloud computing model, a third-party provider delivers hardware and software tools needed for developing or running an application over the internet by hosting the required tools on its own infrastructure. Thus, users don't need to install any specific hardware or software on their computers to run the web-application, which could be a huge benefit for GazeVisual users.

For deployment, a Python based web hosting service [28] was used which provides access to a Linux server where web applications written in Python may be hosted and run publicly. To host the app, Python code for the web application is saved and compiled on the cloud based Python IDE of the hosting web server. All dependencies e.g. libraries like Flask and Dash are installed by setting up a virtual environment on the web-host's portal. The benefit of this hosting service is its support of various Python versions and Python libraries and possibility

to deploy interactive graphics and data analysis facilities through installing required Python packages. Application deployment on this hosting service is also easy and fast.

D. Tests and evaluation

The GazeVisualApp was tested with gaze data in the same way as done with the GazeVisual desktop GUI. Results were found to be identical to those provided by the GUI software. The web app works on desktop and mobile operating systems, with all commonly used web browsers. The GazeVisualApp is live and users may test the app using CSV files from the GitHub repository at: http://gazevisual.pythonanywhere.com/

VII. Utility and Relevance of GazeVisual towards Consumer Electronics Applications

Quality evaluation of consumer devices is a critical aspect that affects both users as well system designers. The users need to know if a device performance meets their needs, while engineers need to learn about the practical limits and capabilities of their systems. Works on evaluating consumer electronic devices like cameras, assistive technologies, biometric systems and wearable sensors are discussed in [34]–[37] which reflects the significance of evaluation methods in consumer electronics use cases.

Evaluating eye tracking data quality is similarly crucial to attain user acceptance and feasibility of using eye trackers, especially in unconstrained consumer applications. It has been shown in [6] that performance (or level of errors) in an eye tracker changes significantly with varying operating conditions. Without a dedicated and easily accessible set of methods for quality evaluation of eye trackers, there is no way for users to know if their eye tracker is delivering optimal performance or if their gaze data is reliably accurate. With uncertain gaze data quality, any gaze based consumer application is likely to fail.

There are several instances where software frameworks have been developed for improving consumer electronics systems in various ways. Examples include [38] where a software framework is described for driver assistance application developments. It reduces software in-homogeneity in presence of a variety of electronic control units so that application developers can work independent of the hardware architecture. In [39] a software framework is presented for product lifecycle information acquisition and management to help in sustainable and environmentally sensitive product design and building intelligent consumer products. In [40] a software development kit is developed to easily and efficiently port complex desktop based software to handheld platforms. A software platform in [41] is designed to allow multiple Internet of Things (IoT) based devices to operate on a single service platform, to enable interoperability across various IoT devices and creation of innovative products.

The GazeVisual software and its web-application fall into such category of software which can be used to improve the quality and service of eye tracking systems that are deployed in a wide range of consumer devices. The software provides multi-criteria evaluation methods for analyzing gaze data and thereby possible sources of errors affecting the data output from eye trackers may be detected. Also it can handle data from different types of commercial eye trackers. GazeVisual (and its web-application) can therefore be highly useful tools for evaluating the quality and reliability of eye trackers implemented in various consumer applications such as handheld devices, desktop and head-mounted systems and also for identifying ways to improve their performance.

A comparison of GazeVisual with a state-of-the art open source (Pygaze) and a commercial [21] gaze data analysis software is shown in Table III. GazeVisual has several capabilities for objective evaluation of eye gaze data that are not present in other software packages for eye trackers.

TABLE III
Comparison of GazeVisual capabilities with other gaze data analysis software

Criteria	GazeVisual	Pygaze [15]	Commercial Software [21]
Gaze accuracy estimation	Yes	Yes	Yes
Gaze error distribution	Yes	No	No
Gaze yaw, pitch	Yes	No	No
Z-score (gaze data scatter)	Yes	No	No
Gaze error 3D plot	Yes	No	No
Live gaze data capture	Yes	Yes	Yes
Stimuli creation	Yes	Yes	Yes
Web-application	Yes	No	No
Cross-platform functioning	Yes	No	No

VIII. Conclusion

In this paper, *GazeVisual*, a freely available software tool capable of analyzing eye gaze data quality based on the raw data output from an eye tracker is presented. This tool enables users to assess and compare their tracker data characteristics under varying test conditions. Several case studies are documented which provide useful working examples for CE engineers on the application of the GazeVisual software for the evaluation of eye-trackers in practical use cases.

A matching web application named *GazeVisualApp* is provided to make the software available via a ubiquitous web interface, and thus easy to use and deployable across multiple platforms. The desktop GUI as well as the web-application is designed to help engineers looking for a common platform for evaluation and comparison of eye tracking systems.

Source code for GazeVisual is available in the GitHub repository which will allow researchers and engineers to use, adapt and introduce improvements to the software in future.

References

[1] A. Poole and L. J. Ball, “Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects,” *Encycl. Human-Computer Interact.*, Idea group, PA, USA pp. 211–219, 2005.

[2] C. H. Morimoto and M. R. M. Mimica, “Eye gaze tracking techniques for interactive applications,” *Comput. Vis. Image Underst.*, vol. 98, no. 1, pp. 4–24, 2005.

[3] J J. Jimenez, D. Gutierrez, P. Latorre, and U. De Zaragoza, “Gaze-based Interaction for Virtual Environments,” *Journal of Universal Computer Science*, vol. 14, no. 19 (2008), 3085-3098.

[4] A. Kar and P. Corcoran, "A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms," *IEEE Access*, vol. 5, pp. 16495-519, 2017.

[5] A. Kar and P. Corcoran, "Gaze Visual - A Graphical Software Tool for Performance Evaluation of Eye Gaze Estimation Systems," *in Proc GEM*, Galway, Ireland, 2018, pp. 1-9.

[6] A. Kar, P. Corcoran, "Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations". *Sensors*. 2018; 18(9):3151.
[7] J. Xu et al. "Real-time eye tracking for the assessment of driver fatigue." *Healthcare technology letters* vol. 5,2 54-58. 31 Jan. 2018,
[8] A. M. Soccini, "Gaze estimation based on head movements in virtual reality applications using deep learning," *in Proc.* 2017 IEEE Virtual Reality (VR), Los Angeles, CA, 2017, pp. 413-414.
[9] J. Weaver, K. Mock, and B. Hoanca, "Gaze-based password authentication through automatic clustering of gaze points," *in Proc.* IEEE Int. Conf. Syst. Man Cybern., pp. 2749–2754, 2011.
[10] H. Lee, et al,"Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.
[11] Q. Lohmeyer, et al, "Toward a new age of patient centricity? The application of eye-tracking to the development of connected self-injection systems", *Expert Opinion on Drug Delivery,* 2018, 16:2, pp163-175..
[12] L. Cornish, et al "Eye-tracking reveals how observation chart design features affect the detection of patient deterioration : An experimental study," *Appl. Ergon.*, vol. 75, no. September 2017, pp. 230–242, 2019.
[13] M. Shojaeizadeh, S. Djamasbi, R. C. Paffenroth, and A. C. Trapp, "Detecting task demand via an eye tracking machine learning system," *Decis. Support Syst.*, vol. 116, no. June 2018, pp. 91–101, 2019.
[14] T. Ujbanyi et al "Pilot Application of Eye-Tracking to Analyze a Computer Exam Test." *Cognitive Infocommunications, Theory and Applications*. Topics in Intelligent Engineering and Informatics, vol 13. Springer, Cham, (2019).
[15] J.Przybyło, E. Kańtoch, P. Augustyniak,"Eyetracking-based assessment of affect-related decay of human performance in visual tasks" *Future Generation Computer Systems,* Vol 92, 2019,Pages 504-515.
[16] O. Špakov, "iComponent - Device-Independent Platform for Analyzing Eye Movement Data and Developing Eye-based Applications." *Dissert. in Interactive Technol.,* vol. 9, University of Tampere (2008)..
[17] A. Tula, A. Kurauchi, F. Coutinho, C.Morimoto. "Heatmap Explorer: an interactive gaze data visualization tool for the evaluation of computer interfaces." *In* Proc. *(IHC '16).* ACM, New York, USA, Article 24, pp1-9.
[18] A. Voßkühler, V. Nordmeier, L. Kuchinke, "OGAMA (Open Gaze and Mouse Analyzer): Open-source software designed to analyze eye and mouse movements in slideshow study designs". *Behav. Res. Methods* 2008, 40, 1150.
[19] E. Dalmaijer,.S. Mathôt, S. V.Stigchel, "PyGaze: an open-source, cross-platform toolbox for minimal-effort programming of eye tracking experiments". *Behavior Research Methods*, 46, 913-921, (2014).
[20] H.Sogo, *Behav Res* (2013) 45: 684, pp. 684–695, 2013.
[21] C. Berger, M. Winkels, A. Lischke, and J. Höppner, "GazeAlyze : a MATLAB toolbox for the analysis of eye movement data," pp. 404–419, *Behav Res* (2012) 44: 404. 2012
[22] S. Tang, R. G. Reilly, C. Vorstius, "EyeMap : a software system for visualizing and analyzing eye movement data in reading," pp. 420–438, *Behav Res* (2012) 44: 420.
[23] Tobiipro.com. (2018). [online] Available at: https://www.tobiipro.com/siteassets/tobii-pro/user-manuals/tobii-pro-studio-user-manual.pdf/?v=3.4.5 [Accessed 2 Oct. 2018].
[24] Andrewd.ces.clemson.edu. (2018). [online] Available at: http://andrewd.ces.clemson.edu/courses/cpsc412/manuals/Gazepoint%20Analysis.pdf [Accessed 2 Oct. 2018].
[25] Eyetracking.com. EyeWorks [online] Available at: http://www.eyetracking.com/Software/EyeWorks [Acc. 2 Oct. 2018].
[26] Y.C. Chen, "A tutorial on kernel density estimation and recent advances," *Biostat. Epidemiol.* 1, 161–187 (2017)..
[27] A. Gibaldi, M. Vanegas, P. J. Bex, and G. Maiello, "Evaluation of the Tobii EyeX Eye tracking controller and Matlab toolkit for research," *Behav. Res. Methods*, vol. 49, no. 3, pp. 923–946, 2017.
[28] S. Popelka, et al, "EyeTribe tracker data accuracy evaluation and its interconnection with hypothesis software for cartographic purposes," *Comput. Intell. Neurosci.*, vol. 2016, 2016.
[29] M. Kassner, W.Patera, A. Bulling. "Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction". *In Proc 2014 ACM Intl. Joint Conf. on Pervasive and Ubiquitous Computing*: ACM, New York, USA, 1151-1160.
[30] J. F. Blinn, "A trip down the graphics pipeline: grandpa, what does 'viewport' mean?," *IEEE Computer Graphics and Applications,* vol. 12, no. 1, pp. 83-87, Jan. 1992 .
[31] Dash.plot.ly. (2018). Dash User Guide and Documentation - Dash by Plotly. [online] Available at: https://dash.plot.ly/ [Acc 9 Nov. 2018].
[32]. P. Vogel, T. Klooster, V. Andrikopoulos and M. Lungu, "A Low-Effort Analytics Platform for Visualizing Evolving Flask-Based Python Web Services," *in Proc 2017 IEEE (VISSOFT)*, Shanghai, 2017, pp. 109-113..
[33] C. Lv, Q. Li, et al, "PaaS: A revolution for information technology platforms," *in Proc 2010 ICENT*, Qinhuangdao, 2010, pp. 346-349.
[34] V. Aarts, et al "Performance of an accelerometer-based pulse presence detection approach compared to a reference sensor," *in Proc 2017 IEEE 14th Intl. Conf. on Wearable and Implantable Body Sensor Networks (BSN)*, Eindhoven, 2017, pp. 165-168.
[35] A Lazarick, P. Wolfhope, "Evaluation of 'non-traditional' fingerprint sensor performance," *in Proc 2016 IEEE Symposium on Technologies for Homeland Security (HST)*, Waltham, MA, 2016, pp. 1-7.
[36] I M. Damghanian, R. Olsson and M. Sjöström, "Performance analysis in Lytro camera: Empirical and model based approaches to assess refocusing quality," *in Proc 2014 IEEE Intl. Conf.on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, 2014, pp. 559-563.
[37] B. F. Smaradottir, S. G. Martinez and J. A. Håland, "Evaluation of touchscreen assistive technology for visually disabled users," *in Proc 2017 IEEE (ISCC)*, Heraklion, 2017, pp. 248-253..
[38] M. Milosevic, et al "Software Platform for Heterogeneous In-Vehicle Environments," *IEEE Trans. Consum. Electron*, vol. 64, no. 2, pp. 213-221, May 2018.
[39] X. Yang et al, "A Component-based Software Framework for Product Lifecycle Information Management for Consumer Products," *IEEE Trans. Consum. Electron*, vol. 53, no. 3, pp. 1195-1203, Aug. 2007.
[40] F. Battaglia, G. Iannizzotto and F. L. Rosa, "An open and portable software development kit for handheld devices with proprietary operating systems," *IEEE Trans. Consum. Electron,* vol. 55, no. 4, pp. 2436-2444, Nov.2009.
[41] J. Yun, I. Ahn, N. Sung, and J. Kim, "A Device Software Platform for Consumer Electronics Based on the Internet of Things," *IEEE Trans. Consum. Electron.*, vol. 61, no. 4, pp. 564–571, 2015.

Anuradha Kar (S'08) Anuradha Kar (S'08) was born in Kolkata, India and received the Bachelor of Technology degree in Electronics & Communication engineering from the West Bengal University of Technology, Kolkata, India, in 2009 and Master of Technology degree in Radio, physics and electronics from University of Calcutta,in Kolkata, India in 2011. She is currently pursuing the Ph.D. degree with the National University of Ireland, Galway.

Her research interests include human computer interactions, application of eye gaze in consumer applications and evaluation of sensor systems.

Ms. Kar won second position at the All India IEEE Student project contest in 2009, and was awarded a two year Master degree fellowship from the Indian Space Research Organization (ISRO) between 2009-2011.

Peter Corcoran (M'95-F'10) received the BAI (electronic engineering) and BA (maths) degrees from Trinity College Dublin, Ireland in 1984. He continued his studies at TCD and was awarded the Ph.D. in electronic engineering in 1987 for research work in the field of Dielectric Liquids.

He is a professor with the Department of Electrical and Electronic Engineering at the National University of Ireland, Galway where he has taught since being appointed to a lectureship in 1986.

He is co-author on 350+ technical publications and co-inventor on more than 300 granted US patents. His research interests include biometrics, imaging, deep learning, edge-AI and consumer electronics.

Professor Corcoran has received numerous industry and academic awards and honors including being elected as an IEEE Fellow in 2010. He is the past Editor-in-Chief of IEEE Consumer Electronics Magazine.

Appendix F: Eye tracking error classification and modelling with machine learning algorithms

Eye tracking error classification and modelling with machine learning algorithms

Anuradha Kar,[a,*] Peter Corcoran,[a]
[a]National University of Ireland, Galway, Department of Electrical & Electronic Engineering, University Road, Galway, H91TK33, Ireland

Abstract. Identifying error characteristics in eye gaze data is a critical task as an eye tracker's accuracy is frequently affected by non-ideal operating conditions in various applications and user platforms. In this work, statistical and machine learning algorithms are used to classify error patterns in gaze data and predict errors produced by an eye tracker when they operate under challenging conditions. For this, gaze data from two platforms, a desktop and a tablet were acquired using an eye tracker when the tracker operated under the influence of several factors (also called error sources), that are known to affect its accuracy-such as variable head pose, user distance and platform orientations. Classifiers were trained on these gaze datasets to identify the impact of different error sources and regression models were used to predict the variability in gaze error levels under the influence of these conditions. The goal of this work is to present methods which may be used to pre-process raw eye tracking data and successively detect and predict the impact of multiple extraneous factors on it. These methods are envisioned to provide in-depth knowledge about an eye tracker's accuracy characteristics and improve quality and reliability of eye trackers operating under unconstrained conditions.

Keywords: Eye gaze, gaze data, eye trackers, pattern recognition, classification, modelling, machine learning, SVM, neural networks.

*****Address all correspondences to Anuradha Kar**,** E-mail: a.kar2@nuigalway.ie

1 Introduction

1.1 Research questions and motivation

Gaze data obtained from eye trackers operating on various consumer platforms is frequently affected by a multitude of factors (or error sources) such as head pose, user distance, display properties of setup, illumination variations and occlusions. The impact of these factors on gaze data are manifested in the form of gaze estimation errors whose characteristics or distributions have not been explored adequately in contemporary gaze research [1]. There are no methods which may be used for analyzing the pattern of gaze estimation errors and investigating further details about their occurrences. For example, it is not known whether the above factors produce any particular pattern of errors or if the nature of gaze errors follows any statistical distribution

or if they are simply random. These aspects cannot be understood by looking at raw gaze data which is almost always corrupted with noise and outliers, or even by studying mean error values.

With respect to gaze error analysis, several questions arise: These include: 1) how can gaze errors caused by one error source be distinguished from those caused by another 2) how can the presence of different error sources in a certain gaze dataset be detected without prior knowledge 3) is it possible to predict the level of gaze errors that might be caused by different error sources. 4) can suitable features be extracted from gaze datasets to identify the different error sources. These questions form the main topics that are addressed in this paper.

This paper focuses on defining methods for detailed analysis of eye tracking data obtained from a generic commercial eye tracker for the detection, identification and prediction of gaze errors. The aim of this work is to observe gaze error patterns produced by three error sources which commonly affect eye trackers in static and dynamic platforms like desktop and tablets. These include head movement, user distance and platform orientation. It was found that there currently exists no publicly available gaze dataset which contains raw gaze and ground truth data along with signatures of these error sources. Therefore a new eye tracking dataset was built by collecting gaze data from 20 subjects using a commercial eye tracker on both a desktop and a tablet platform. This dataset may be obtained from the authors on request and is also to be published in an open repository for use by the gaze research community (link to the dataset webpage is in Section 5 below). During data collection, variations in head pose, user distance and platform orientations were introduced sequentially and in a calibrated manner so that gaze data contains the influence of one known condition (or error source) at a time. Reference data, which does not have the influence of any of these conditions, was collected as well.

For detecting gaze error patterns caused by the different error sources, the collected data is used for training machine learning (ML) algorithms by creating training features from the datasets. As the operating conditions change, the feature variables in the training data get affected and as a result the ML models learn to differentiate between error patterns caused by different sources. In this way, anomalous gaze data may be distinguished using classifiers which have been trained using both affected and reference gaze datasets. Finally, regression algorithms are built to model and predict gaze errors which may be produced by above three error sources.

1.2 Background and scope

Pattern classification and modelling approaches have been applied on eye tracking data for various purposes. For example, in [2], the effect of video frame rate on viewing behavior and visual perception of users is studied by comparing gaze data patterns collected for high and low video frame rates. In [3] a new hybrid fuzzy approach is introduced to distinguish gaze patterns when users perform face and text scanning. In [4], dominant gaze characteristics of experienced and inexperienced train drivers are classified using Markov Cluster (MCL) algorithm, and marked differences are observed between gaze patterns of the two driver classes. [5] uses a Bayesian Mixture Model to learn the gaze behavior of drivers who perform a variety of tasks during conditionally automated driving, by classifying the driver's fixations and saccades. Gaze patterns are used with clustering and classification algorithms to predict user's intention to perform a set of tasks in [6].

A special de-noising and segmentation algorithm based on Naive Segmented Linear Regression and Hidden Markov Models is developed in [7] to classify gaze features into fixations, smooth pursuits, saccades and post-saccadic oscillations using good quality as well as noisy data.

Automated classification of fixations and saccades were achieved using random forest classifiers on high quality as well as noisy gaze datasets in [8].

It may be observed that gaze or eye movement pattern modelling are mostly applied towards either cognitive studies, i.e. to interpret viewing patterns or distinguish between oculomotor event types e.g. saccades, fixations and smooth pursuits. With respect to studying gaze error patterns, however only a few works are reported. Examples include [9] which estimates the two-dimensional gaze error distribution and uses a predictive model to shift gaze according to a directional error estimate. This is based on a previous work [10] which proposed a gaze estimation error model that can predict gaze error by taking into consideration factors such as mapping of pupil positions to scene camera coordinates, marker-based display detection, and mapping from scene camera to on-screen gaze coordinates. However, no works till now have covered the aspect of classification of gaze error patterns induced by different operating conditions and prediction of error levels under their influence, which is presented in this paper.

In this work, the term gaze error is frequently used which is the angular difference between the gaze locations estimated by an eye tracker and actual locations of the visual targets, typically appearing on a display screen or viewing area. Gaze error patterns in this work signify magnitude distributions of the gaze error values over a given number of samples or display area. Error sources refer to non-ideal operating conditions such as high degree of head pose variation with respect to frontal pose, too long or too short user distances or conditions where an eye tracking platform is tilted to different angles as opposed to their neutral positions. Such conditions are considered in this work since they occur frequently during practical operations of eye trackers and significantly affect an eye tracker's accuracy, but the patterns of error induced by them on gaze data characteristics are unknown till now.

In this work, the impact of these error sources on the gaze error patterns produced by a given eye tracker is studied. The scope is limited to studying one error source or operating condition at a time (i.e. either head pose or user distance or platform pose is varied at a time). However eye trackers could face more complex scenarios, for example two or more error sources could occur together and affect an eye tracker's data. Such analysis is not covered in this work as firstly relevant data is not available, and secondly occurrences of such complex circumstances are rarer than the ones considered here. It is likely that at a time there is only one error source affecting an eye tracker with major influence compared to other sources which may or may not be present. For example, user distance is fixed in a desktop or automotive based eye tracking systems and head motion is the major source of gaze tracking error in these platforms. Our analysis is based on data collected from static remote eye tracker setups on desktop and tablet platforms. This study is done with participants not wearing glasses, in indoor space and under uniform illumination levels, to rule out the impact of occlusion and illumination changes on gaze data

1.3 Organization of contents

The paper is organized as follows. Section 2 describes the gaze data collection setup and procedure and steps for data pre-processing and exploration. Section 3 provides details about the gaze error detection methods using machine learning models and Section 4 presents the various regression algorithms used in this work for modelling errors produced by different conditions.

2 Experimental methodology and data exploration

The concepts of this work have been implemented in several phases. Eye tracking data was collected through special experiments using two consumer platforms, a desktop and a tablet, under different operating conditions. The collected data went through the processing pipeline

(Fig. 1) with initial investigation using statistical methods and visualizations before being fed to the machine learning models in the final phase. Data collection experiments, experimental setup and steps of the data analysis workflow are described below.

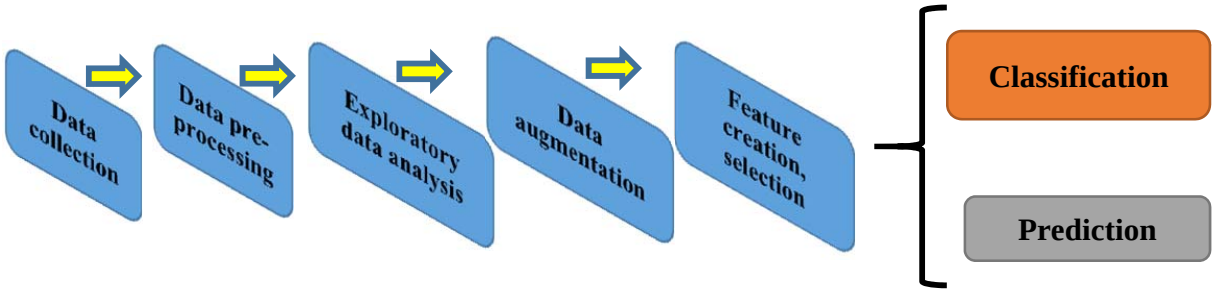


Fig.1 Data processing and analysis pipeline followed in this work

2.1 Eye tracking data collection

A detailed description of the gaze data collection process and setup using a commercial eye tracker has been provided in our previous work [1] in its Section 3.1. For the current work, data was collected using the same methodology but from a larger group of 20 participants and for four user distances from both the desktop and tablet platforms.

2.2 Eye tracking UI and device

For eye tracking data collection, an eye tracker coupled with a visual stimulus interface (also called UI) is used. The trackers used are a Tobii EyeX 4C and an Eye tribe (for pilot data collection) remote eye tracking device which come with their own calibration routines. During an experiment session participants are seated in front of the tracker (Fig.3) with a chin rest and the UI runs simultaneously with the eye tracker. The UI shows a moving dot which sequentially traces a grid of (5x3) known locations (also called area of interest or AOI) over the desktop or tablet's display (Fig 3 right). The dot radius is 10 pixels and it stops at each AOI for 3 seconds

before moving to the next and gaze data is collected while participants look at the dot. Gaze data comprises of a participant's gaze coordinates (x, y positions in pixels) on the (desktop/tablet) display and corresponding time stamps as estimated by the tracker. The known on-screen dot locations form the ground truth data and are used for accuracy calculations. Chin rest is used for stabilizing a participant's head for all experiments.

2.3 Setup and experimental details

The desktop setup consists of the eye tracker mounted on the screen of a desktop computer (Fig 3, top). The screen diagonal is 22 inches, with a pixel resolution of 1680 x1050. Two different experiments are done with this setup. These are: (a) User distance experiments: In these, gaze data is collected at user-eye tracker distances of 50, 60, 70 and 80 cm. (b) Head-pose variability experiments (Fig 2a): Head pose here refers to the position of a user's head in 3D space in terms of roll, pitch and yaw (RPY) angles. During the experiments, a user is seated at a fixed distance (60 cm) from the tracker and is asked to vary their head position to fixed but distinct pose (RPY) angles each time, with respect to the frontal position (RPY=0) while looking at the UI on the display. Their gaze is tracked on the UI and their head position is tracked simultaneously using an active appearance model [11], that measures head pose RPY angles with 1 degree accuracy. It may be noted that neutral head pose gaze data is the same as user distance gaze data at 60 cm.

For the tablet setup, two experiments were implemented with the same test UI as used for the desktop. The first ones were the user distance experiments done in the same way as for the desktop platform, described above. The other experiment was done for studying the impact of variable platform orientation on eye tracker data (Fig. 2b). For this, the tablet device was first mounted with the eye tracker and the orientation of this combined setup was varied to known

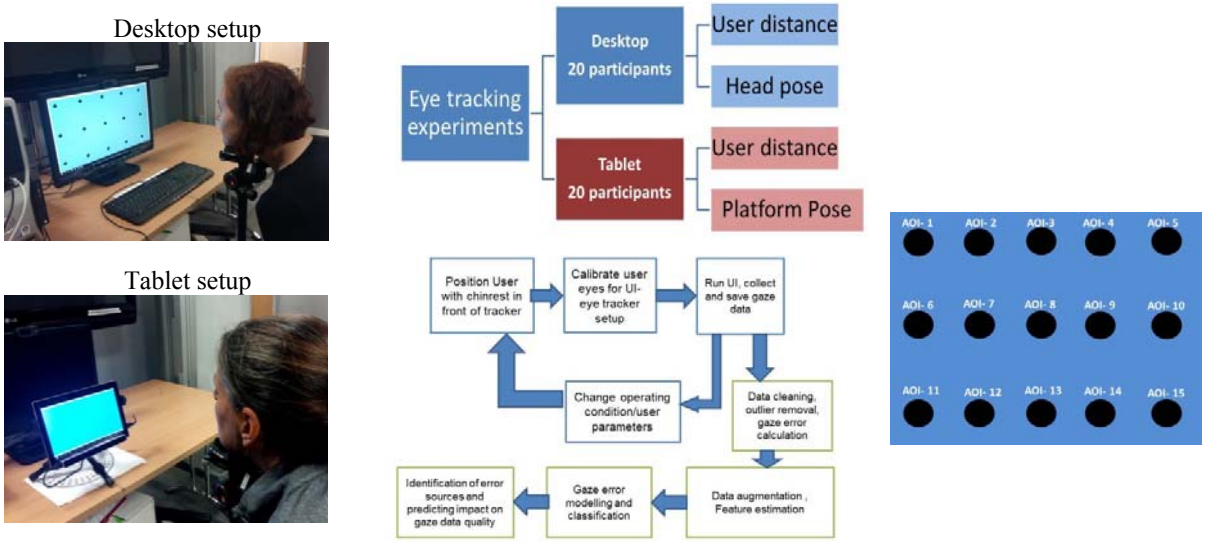
platform roll, pitch and yaw angles (20 degree in each of roll, pitch yaw directions). The user's head and distance from the tablet-tracker setup were kept fixed (Fig. 3). Eye tracking data was collected for each tablet orientation (at fixed user-tracker distance of 60 cm). Data for the neutral tablet orientation is the same as gaze data at 60 cm from the tablet user distance experiment. The tablet screen has a diagonal size of 10.1 inches and pixel resolution of 1920 x800.

Maximum head and platform poses allowed by the eye tracker is 20 degrees and no tracker data can be obtained at distances below 50cm or above 80cm. Data from user distance experiments are called UD50, UD60,UD70, UD80 for distances of 50,60,70, 80 cm. "HP" is used to denote head pose experiments. The same group of 20 participants (15 male, 5 female) were involved in data collection for all experiments.

(a) (b)

Fig 2. (a) Face model showing different head poses in real time (b) different platform poses of tablet with tracker [1]

The process flow of the experiments comprised of first positioning the user in front of the desktop or tablet setup and calibrating their eyes with the calibration software provided for the eye tracker. Next, the UI described above is run along with its data-logging routine to record eye tracking data in comma separated values or CSV files, along with millisecond timestamps. Data collected from these especially designed experiments is then used for processing and analysis.

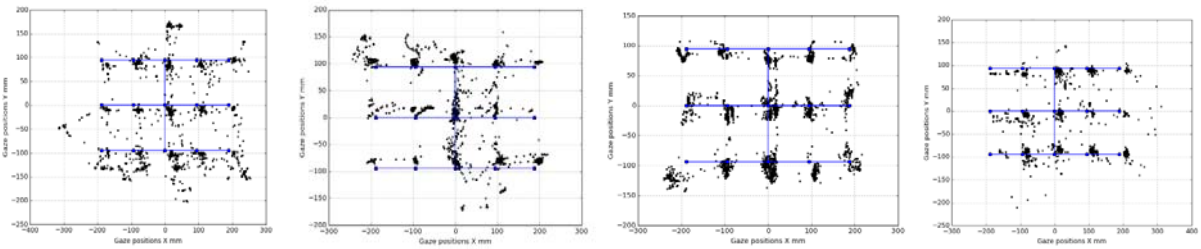


(a) (b) (c)

Fig.3 From left a) Experimental setups for desktop and tablet platforms b) (top) list of experiments performed in this work (bottom) process flow for experiments, data collection, analysis c) user interface for gaze data collection

2.4 Data preparation

Figure 4 below shows the raw gaze data overlaid on ground truth, as obtained from the different experiments described above. It can be seen that it is impossible to decipher error patterns from gaze data by simply looking at it in raw form and data from very different operating conditions often look similar and vice versa. However, as will be shown next, eye tracking data obtained under different operating conditions do have diverse error distributions and statistical properties.



(a) (b) (c) (d)

Fig. 4 Raw gaze data (black points) overlaid on ground truth data(blue lines) for (a) neutral head pose (2)head pose with roll of 20 degrees (c), head pitch of 20 degrees (d) head yaw 20 degrees. It can be seen that it is very hard to visually distinguish between data from different experiments done under different conditions, unless special pre-processing steps and learning algorithms are employed to identify the error source affecting the eye tracker.

This subsection describes the steps of preparing the gaze datasets to bring them into a suitable form before applying machine learning models. The first step to prepare the gaze data for the learning tasks is to convert the raw gaze data coordinates (in pixels) into frontal gaze angles and gaze yaw and pitch angles. The raw gaze x,y pixel coordinates of the left and right eye (X_{left}, Y_{left} & X_{right}, Y_{right} respectively) obtained from the tracker are used to estimate gaze angle and gaze yaw, pitch angles as follows [1]:

$$\text{GazeX} = \text{mean}\left(\frac{X_{left} + X_{right}}{2}\right), \text{GazeY} = \text{mean}\left(\frac{Y_{left} + Y_{right}}{2}\right) \quad (1)$$

The on-screen distance (OSD) of a user's gaze point is the distance between the origin and a certain gaze point with the coordinates (GazeX, GazeY). In our case, the tracker is attached directly below the screens and the origin is the center of the screen. Therefore:

$$\text{OSD (mm)} = \mu\sqrt{((\text{GazeX})^2 + (\text{GazeY})^2)} \quad (2)$$

Where μ is the pixel pitch of the display where gaze is tracked in units of mm/pixel.

The gaze angle of a point on screen relative to a user's eyes is calculated as:

$$\text{Gaze angle } (\theta gaze) = \tan^{-1}(\text{OSD}/Z) \quad (3)$$

The ground truth (θgt) gaze angles for the AOI locations with coordinates (AOI_X, AOI_Y) are:

$$\text{OSD\_GT} = \mu\sqrt{((\text{AOI\_X})^2 + (\text{AOI\_Y})^2)}$$
$$\text{Ground truth angle } (\theta gt) = \tan^{-1}(\text{OSD\_GT}/Z) \quad (4)$$

The gaze yaw and pitch angles are derived as follows:

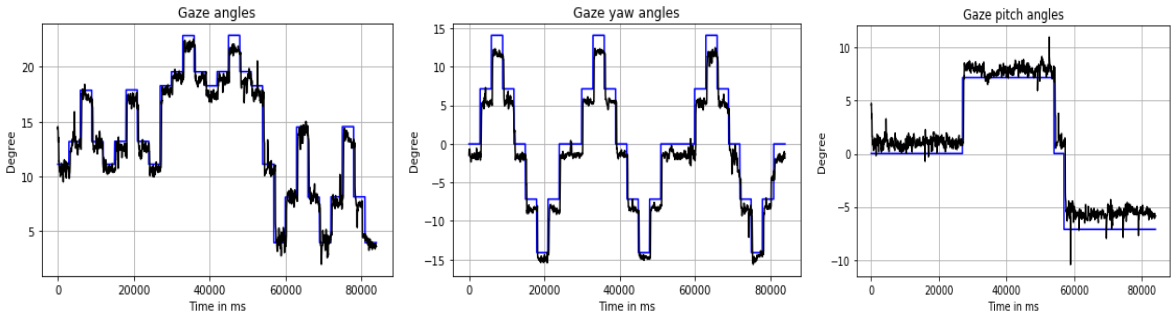
$$\text{Gaze pitch}(\theta pitch) = \tan^{-1}(\text{GazeY}/Z), \text{Gaze yaw } (\theta yaw) = \tan^{-1}(\text{GazeX}/Z) \quad (5)$$

The ground truth pitch and yaw angular values for each AOI dot with screen coordinates (AOI_X, AOI_Y) are given by:

$$\text{AOI pitch} = \tan^{-1}(\text{AOIy}/Z), \text{AOI yaw} = \tan^{-1}(\text{AOIx}/Z) \quad (6)$$

These gaze variables along with their statistics would later be used to construct the feature vectors for the learning algorithms. The plots of gaze frontal and rotational angles are shown below in Fig. 5. Throughout this paper, the terminologies gaze angle, gaze yaw and gaze pitch

angle will be used to indicate the variables defined by Eqns 3-6 above. Readers must not be confused between the terms gaze yaw/pitch and user head roll/yaw/pitch and platform pose roll/yaw/pitch which are also used in this paper.



(a) **(b)** **(c)**

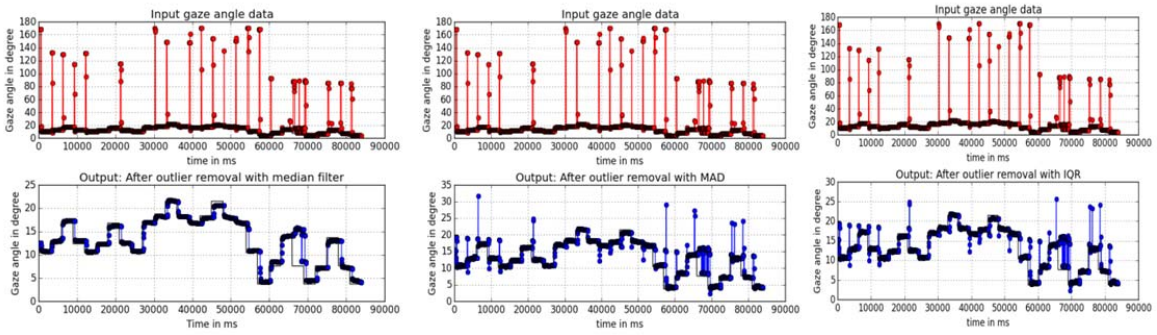
Fig. 5 Time series of a) gaze angles b)gaze yaw and c) gaze pitch angles (black lines, along with ground truth- in blue lines) for one person during one experimental session for neutral pose captured on the desktop setup

The common problems associated with analyzing the collected dataset were: a) presence of outliers b) missing or null values c) unequal data row lengths. Therefore, before using the data to train the models and yield meaningful results, it was essential that collected data goes through certain pre-processing steps. For filling the missing values, a mean substitution was used. For outlier removal, the following methods were tested i) 1D Median filtering: This method can detect isolated out-of-range values from legitimate data features. In this method, value of a data point is replaced by that of the median of all data points in a neighborhood w [12] such that:

$$y[m,n] = \text{median } \{x[i,j], (i,j) \epsilon w\} \quad (7)$$

ii) Median absolute deviation: This is calculated by taking the absolute difference between each point and the median, and then calculating the median of those differences. This is more robust than using standard deviation for outlier detection as standard deviation is itself affected by presence of outliers [13]. c) IQR: The concepts of using Z-score and inter-quartile range (IQR) for studying outliers have been discussed in our previous work [1]. A data point is denoted as an outlier if value for the point is 1.5·IQR above the third quartile or below the first quartile of the

data. Results from the three outlier removal methods are shown in Fig.6 a-c and it was found that median filtering (with a kernel size =41, which is the mean number of data points around each AOI) worked best for all of the datasets..



(a) (b) (c)

Fig. 6 Outlier detection and removal with a) median filtering b)MAD and c) IQR methods. Median filtering is seen to remove nearly all outliers.

2.5 Exploratory data analysis and visualizations

Since the eye tracking datasets were collected under unconstrained conditions and the nature of such data or its distributions are unknown, data exploration was essential to observe any underlying patterns before proceeding to the learning step. This is done after the data cleaning steps described above as the presence of outliers and null values made it impossible to study any meaningful patterns prior to data preprocessing. After null and outlier removal, gaze angular errors for frontal and rotational components are computed in absolute units (degrees) as the angular deviation between ground truth and estimated gaze angular values using Eqns 3-6, as:

$$(\text{Gaze frontal angular error})_i = (\theta\text{gaze})_i - (\theta\text{gt})_i \quad (8)$$

$$(\text{Gaze yaw angle error})_i = (\theta\text{yaw})_i - (\text{AOI\_Yaw})_i \quad (9)$$

$$(\text{Gaze pitch angle error})_i = (\theta\text{pitch})_i - (\text{AOI\_Pitch})_i \quad (10)$$

The gaze angular error and gaze yaw, pitch errors are three categories derived from the same gaze data sample. This aspect will be used later in generating the training sample set from the collected gaze datasets.

2.5.1 Studying gaze data statistics for desktop and tablets

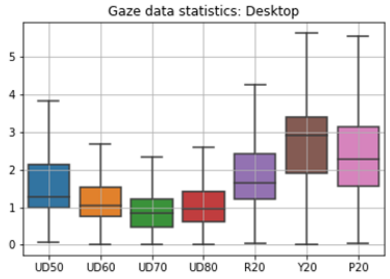
The first step to studying error characteristics of gaze data obtained from the different experiments is to look into the error statistical parameters such as mean, median absolute deviation, inter-quartile range and 95% confidence interval. These parameters will also form parts of the feature vector for the classification studies later in this paper. Fig 7a and Table I show the statistical properties of gaze errors for different desktop experiments. It is seen that gaze error is higher at low user distances and error reduces as user-tracker distance increases. This is primarily due to reduction in visual angle and eccentricity with increasing distance as discussed in [1]. Error due to head yaw is seen to have the highest magnitude although errors due to head pitch have the highest inter-quartile range-or highest variability in error values. Also error levels due to various head poses are quite higher compared to when head pose is neutral (UD60 values in Table I). All values in the tables have units in degrees of angular resolution.

Table 1. Gaze error statistics from desktop experiments

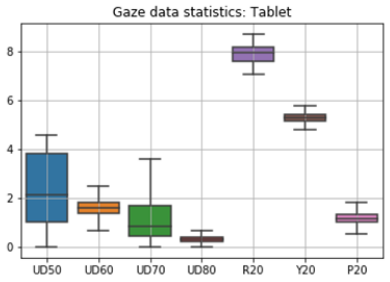
	UD50	UD60	UD70	UD80	Roll 20	Yaw 20	Pitch 20
Mean	3.37	2.04	1.21	1.02	3.7	8.51	3.15
MAD	3.49	1.77	0.82	0.66	3.63	10.0	1.90
IQR	1.13	0.77	0.76	0.79	1.21	1.49	1.59
95% interval	3.15-3.59	1.90-2.18	1.15-1.26	1.16-1.24	3.30-4.09	7.60-9.43	2.83-3.47

Table 2. Gaze error statistics from tablet experiments

	UD50	UD60	UD70	UD80	Roll 30	Yaw 30	Pitch 20
Mean	2.68	2.46	0.59	1.55	7.74	4.25	2.45
MAD	0.38	0.42	0.29	0.24	0.77	0.60	0.46
IQR	0.39	0.54	0.33	0.22	0.75	0.53	0.23
95% interval	2.65-2.71	2.43-2.48	0.57 -0.61	1.53-1.57	7.69-7.80	4.22-4.29	2.41-2.49



(a)



(b)

Fig 7 Boxplot of gaze errors from (a) desktop b) tablet experiments

Fig 7b and Table II show the statistical properties of gaze errors for different tablet experiments. The magnitudes of errors due to tablet pose changes are high and the highest error is caused due to platform roll variations. It is also seen that the error characteristics from tablet data are quite different than those from the desktop platform. Compared to desktop data, the error magnitudes are lower for tablet for all user distances. Also magnitude of errors produced due to different platform poses (Fig 7b) are higher compared to errors induced due to head pose (Fig 7a).

2.5.2 Studying gaze error distributions for desktop and tablet data

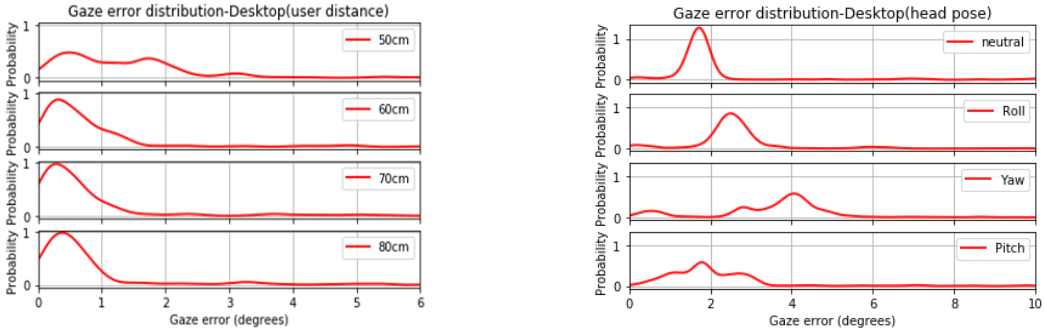
The 1-dimensional distributions of the angular error values for different gaze datasets are studied using Kernel Density Estimate (KDE) [14]. Since the exact distributions of the gaze errors are unknown, Kernel density estimation is useful since it is a non-parametric way to approximate the probability density function of the data, compared to parametric estimation where a fixed functional form and its parameters are required to fit the data. For data with samples x(i), using a kernel function (*K*) and bandwidth h, the probability density at a point x is :

$$\text{KDE} = \sum_{i=1}^{N} K((y - x_i)/h \qquad (11)$$

The Gaussian kernel K is given by

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-0.5u^2} \qquad (12)$$

Data from user distance experiments (50-80 cm) are used to create the KDE of gaze errors in Fig 8a. KDE plots of gaze errors for neutral and head roll, pitch, yaw directions are shown in Fig 8b.

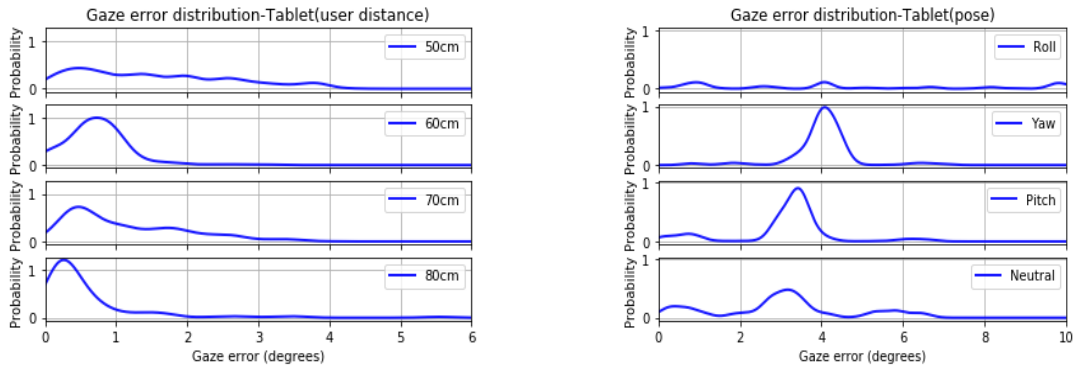


(a) (b)

Fig 8 Gaze error distribution due to: a) user distance (b)due to head pose variations on desktop platform

It may be seen from Fig 8a and b that each operating condition (e.g. user distance or head pose) leaves a definite signature on the gaze error distributions. As an example, for user distance experiments, Fig 8a shows that clear distinction exists between patterns of gaze errors for different user distances (top-bottom plots correspond to user distances of 50, 60, 70, 80 cm) as the error distribution shifts towards lower error values when user distance increases. The reason behind this has been discussed above in Section 2.4.1. Also, Fig 8b shows that head pose deviations from the neutral position induce different pattern of errors in for different pose directions. However, the error patterns induced by these factors are difficult to decipher when looking at raw gaze data or simple error magnitudes.

Using the data collected from tablet experiments, the KDE plots of gaze errors for four different user distances and different platform poses are plotted below. It is again seen that different operating conditions have their individual impacts on the gaze error distribution. Further, the gaze error KDE plots (Fig 8, 9) are found to be non-Gaussian or resembling any known statistical distribution function, which makes it difficult to predict the nature of gaze errors produced by different error sources. These aspects forms the background for studying gaze error characteristics due to different error sources and implementing the learning tasks described next.



(a) (b)

Fig 9 Gaze error distributions due to (a) different user distance from tablet (b) tablet poses.

2.5.3 Studying spatial error distribution properties

2D spatial distribution of gaze error values over the display screen area, as a function of corresponding visual angles for different datasets are computed and displayed. These plots show that factors like user distances and head pose angles have varied impact on gaze error spatial patterns, with minimum errors obtained at the center of the display, for neutral head positions. Similar features are seen from tablet data plots, for different user distances and platform poses.

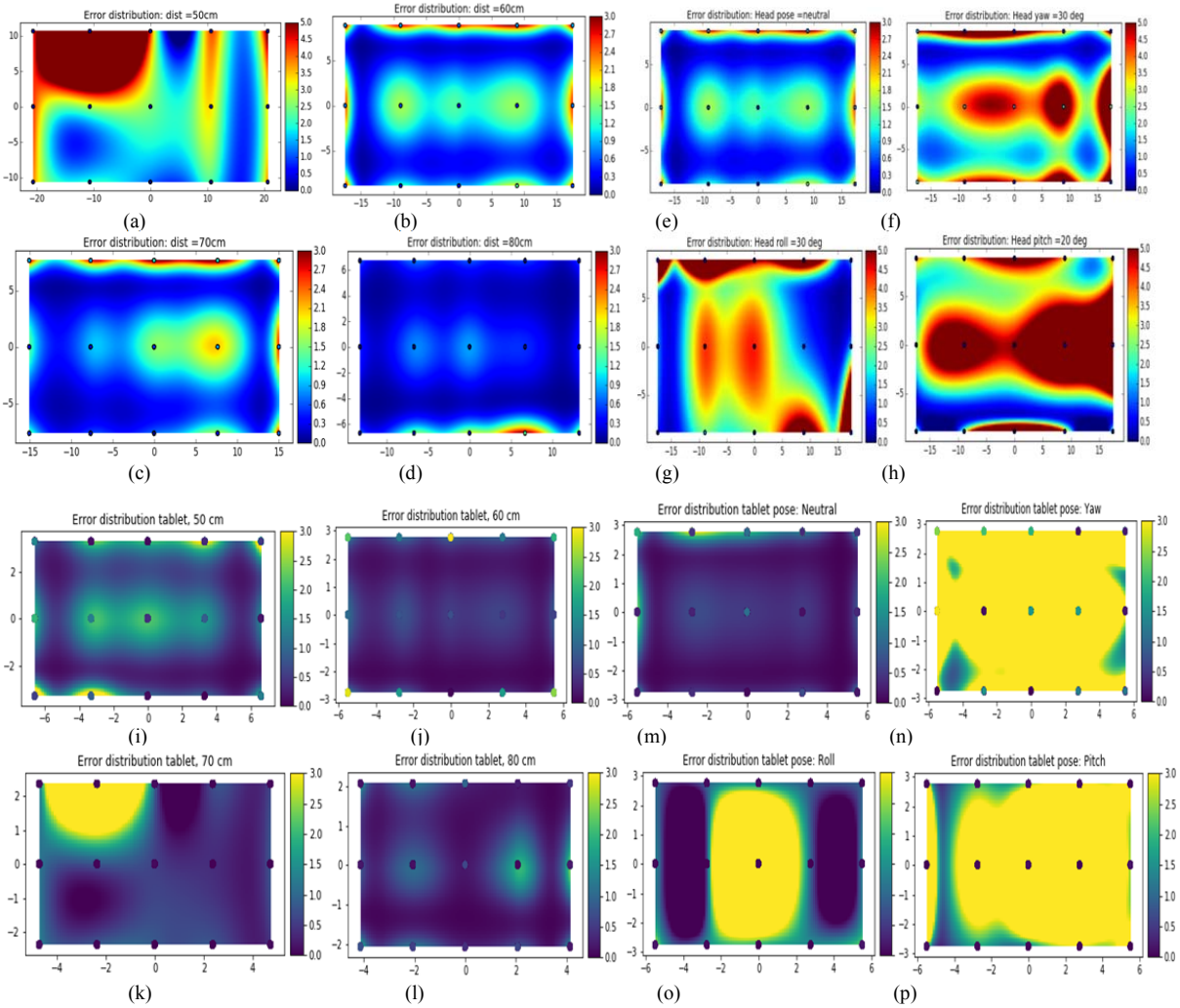


Fig 10 Gaze error spatial distribution as a function of visual angles (x axis= yaw angle, y axis= pitch angle) over display due to user distance for desktop (Fig 10 a,b,c,d) and tablet (Fig 10i,j,k,l). Fig 10 (e,f,g,h) show spatial error distributions due to head pose variations on desktop, Fig 10 (m,n,o,p) show error distributions due to tablet poses.

2.5.4 Studying data correlations

Desktop data from user distance experiments at 50, 60, 70 and 80 cm (UD 50, UD60, UD70, UD80) and head pose experiments at head roll, pitch, yaw angles of 20 degrees,(R20, Y20, P20) are used to compute desktop data correlation matrix of Fig 11a. Data from tablet experiments for different user distances and platform poses are used to compute the tablet data correlation matrix of Fig 11b. As can be observed, gaze data collected under different operating conditions from the same platform and eye tracker do not have any correlations between their characteristics.

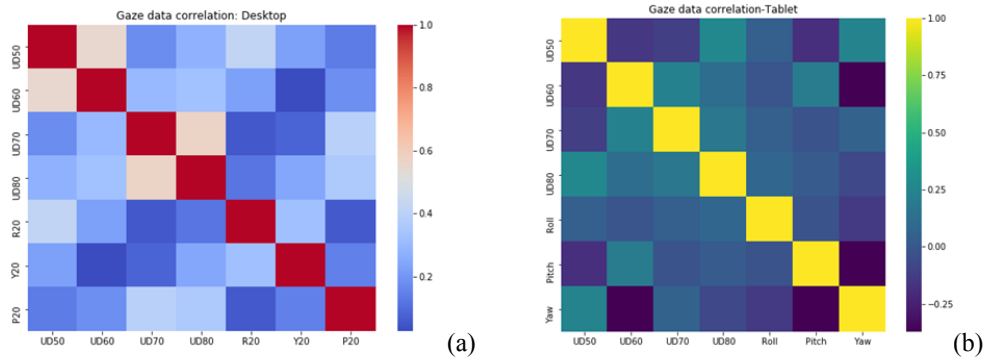


Fig.11 Correlation between data collected from a) desktop (R20, Y20, P20 refers to head pose) (b) tablet

2.5.5. Discussions

In this section, various parts of the gaze data processing pipeline were described and detailed visual and numerical exploration of eye tracking data was done. The gaze data collected from different experiments showed varied levels of in-homogeneities and only after outlier removal, distinct gaze error patterns could be observed which will be used in the next sections for the learning models. One significant aspect noted is that gaze errors from tablet are much lower than errors from the desktop for the same user distances. This could indicate that the distinguishing aspect between the two platforms, which is display size and resolution could be a factor determining the error levels for an eye tracker. However, further comparison of data from the two platforms is not done in this work. Other takeaways from this section include identification

of robust outlier removal methods for gaze data and observations on gaze error distributions which are heavily affected by operating conditions but are not observable in raw gaze data plots. Also studies on the correlation of different eye tracking datasets (Fig 11) reveal an important aspect, that under different operating conditions an eye tracker's data may behave in totally independent ways which are not related to the data characteristics under stable conditions.

3 Identification of error patterns in eye gaze data

3.1 Objectives and task definition for classification of desktop and tablet data

In this section, the main goal is to identify a certain error source or operating condition solely from the output data from an eye tracker which has been influenced by the condition. For this purpose, multiple classifier models are trained using data collected from the different eye tracking experiments, which are seen to produce different error patterns. The objective is to see if machine learning models [15][16] can learn to distinguish between these gaze error patterns as they appear among a mix of data captured under different operating conditions.

The following classifications tasks are performed using the desktop datasets: 1) classification of errors for different user distances (i.e. between 4 classes of data from user distance- 50, 60, 70, 80 cm datasets) (2) classification of errors for different head poses (i.e. between 4 classes : neutral pose, roll (20 degrees), pitch (20 degrees), yaw (20 degrees) datasets), (3)classification between head pose and user distance errors patterns (i.e. between 4 user distance classes, and 3 head pose classes, total 7 classes). With data from tablet experiments, the following classification tasks are implemented: 1) classification of errors for different user distances (i.e. between 4 classes of user distances data) (2) error classification for different tablet orientation poses (i.e. between 4 classes of data from neutral tablet pose, roll (20 degrees), pitch (20 degrees), yaw (20

degrees) datasets), (3)classification between tablet orientation and user distance errors patterns (i.e. between 4 user distance classes, and 3 tablet pose classes, total 7 classes).

Before implementing the classification algorithms on both desktop or tablet data, the user distance, head-pose and tablet orientation datasets are augmented to increase number of training samples[17][18]. After this, training features are constructed and formatted before input to the models. Same data augmentation strategies are used for both desktop and tablet data. For training and testing, gaze angle, yaw, pitch feature datasets are created and used. Classification results from desktop data are in Section 3.3 while that from the tablet datasets are in Section 3.4.

3.2 Data augmentation strategies

With our dataset size (20 persons x 4 operating conditions each for user distance and head pose), in order to use sufficient number of features for classification without facing overfitting problem, augmentation of the dataset was essential. In this work, 10-fold augmentation strategies were used on the gaze angle, yaw and pitch error datasets estimated from raw data using Eqns 8-10. The methods used for data augmentation are as follows: 1) Adding Gaussian noise: Gaze error magnitudes at all data points are perturbed with Gaussian noise with 0 mean and 0.2 sigma [19] 2) Adding jitter: Human eye jitter is modelled as pink noise [20][21] ,in which the power spectral density is inversely proportional to the frequency of the signal, given by PSD= $1/f^{\alpha}$. The jitter signal is simulated by first generating white noise with mean, sigma (0, 0.2) and successively applying a pink noise filter with the parameters α= 0.8 at a frequency of 2Hz and added to the error datasets. 3) Interpolation: Linear interpolation on the input data points is used to produce variants of the original gaze error samples [22]. 4) Convolution: The error signals are convolved with a raised cosine kernel of window size N=30 with the form of Eqn 13 below to produce smoothed variant of the original data [23].

$$w(n)=\frac{1}{2}\left(1-\cos\left(\frac{2\pi n}{N-1}\right)\right) \quad (13)$$

The convolution operation is given by: $(f*g)(t)\triangleq\int_{-\infty}^{\infty} f(t-\tau)\,g(\tau)$ (14)

5)Time shifting: The gaze data are shifted by 10 samples to estimate new variants in error values from the shifted datasets [24] 6) Combinations: Combinations of adding the different noise patterns to interpolated signal are used to augment the dataset. 7) Flipping: Horizontal and vertical flipping of error magnitudes at the different AOIs are used to augment the dataset as well as to remove any viewer bias towards the top, bottom and side locations of the screen. In horizontal flipping, the error magnitudes of top AOI-1 to AOI 5 (Fig 3) are replaced by the values of bottom AOIs (AOI-11 to AOI 15). In vertical flipping, error magnitudes of left AOIs (AOI No. 1,6,11) are swapped with right AOIs, (No. 5,10,15). Fig 12 shows how the samples from a dataset are modified by these augmentation methods. Table 3 shows how training samples are expanded from the collected dataset.

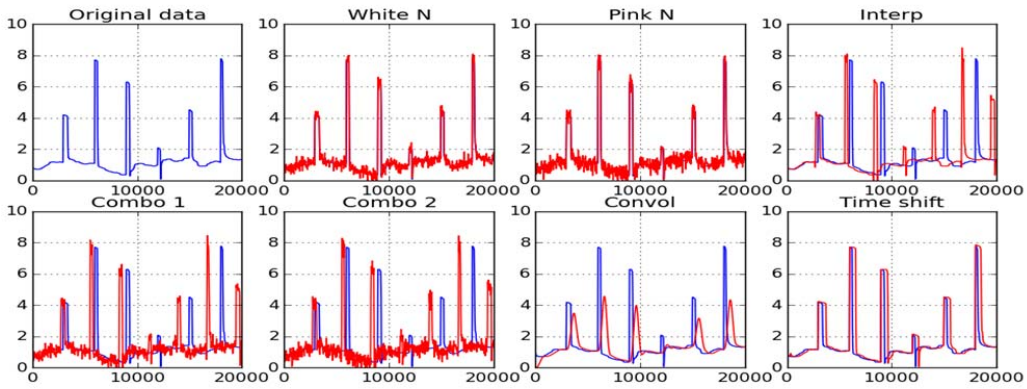


Fig 12. Shows a sample from a dataset after applying each augmentation strategy. The blue lines represent samples from original data, while the red lines are from the augmented dataset after applying each strategy.

3.3 Feature engineering, exploration and selection

The original data sample set in this work has 20 participants and for each subject, there are three categories- i.e., gaze angular error, yaw error and pitch error (Eqns 8-10) from which training

features are computed[25]–[28]. The training feature set is constructed by estimating error magnitudes at 15 AOI locations distributed all over the display screen as shown in Fig 3 above (to account for the spatial distribution of errors), and statistical values for each sample, i.e. mean error (μ), error standard deviation (σ), interquartile range (IQR) and upper and lower bounds of the 95% confidence interval of the sample. Thus each training sample has 20 features as follows:

$$[\text{Error\_AOI-1, Error\_AOI-2...Error\_AOI-15, } \mu, \sigma, \text{ IQR, 95\% conf upper, 95\% conf lower}]_{\text{sample}} \quad (15)$$

The above feature set is calculated from gaze angle as well as yaw and pitch angle data for each sample. For training the machine learning models, the features from all datasets are standardized so that they have zero mean and unit variance and shuffled randomly before splitting the datasets into train and test sections. The proportion of train vs test samples are varied between 0.4 to 0.25. Table 3 below describes the contents of all the training and test datasets used in this work.

For visualizing the high dimensional feature set, the t-Distributed Stochastic Neighbor Embedding (t-SNE) method [29]is used , which maps data points x_i in the high dimensional feature space R^D (D is the dimension of the feature set, D=20 here) to points y_i in a lower d-dimensional space (R^d , here d= 2) by finding similarities (p_{ji}) between the data and learning the corresponding low dimensional mapping points $y_1, \ldots, y_N$ (with $y_i \in R^d$ that reflects these similarities as best as possible[30]–[32]. The pairwise similarity (p_{ji}) between points x_i and x_j is:

$$p_{ji} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2/2\sigma_i^2} \quad (16)$$

For observing the training data in feature space, the t-SNE algorithm (with n_components=2, perplexity=80) is applied on the feature sets computed from user-distance, head-pose and platform pose datasets and plotted in Fig 13 a and b for desktop and tablet data respectively. On datasets from both desktop and tablet, the t-SNE points look scattered with no structure. For head pose data and platform pose data, and mixed datasets, the t-SNE plots show several

clusters, but within them the class labels are found to be highly mixed. This reflects the high degree of complexity of the datasets used in this work since the different classes within them do not form any observable clusters, neither are symmetrically distributed nor have any clear separation between them. This is close to real world gaze datasets in which normal gaze data is most often mixed with anomalous data occurring due to unpredictable operating conditions.

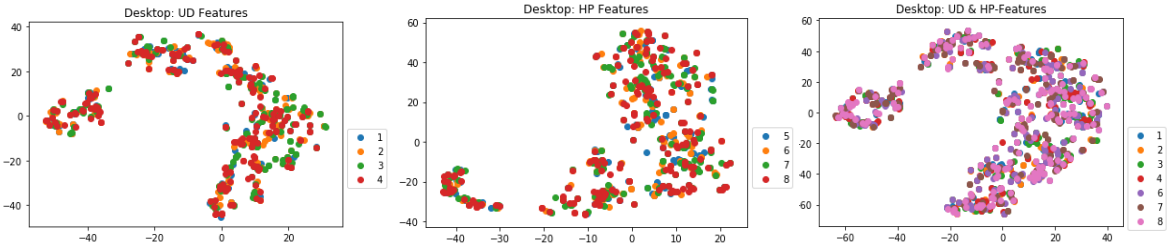


Fig 13a. t-SNE plots for (i) user distance (ii) head pose (c) merged user-distance & head pose for desktop datasets.

Legends 1, 2, 3, 4 are user distance classes UD50-UD80, 5,6,7,8 are head pose classes neutral, roll, pitch, yaw

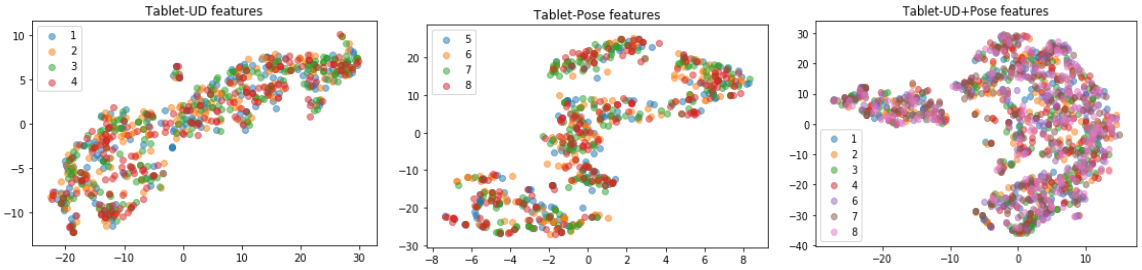


Fig 13b. t-SNE plots for (a) user distance (b) tablet pose (c) merged user-distance & tablet pose for tablet datasets.

Legends 1, 2, 3, 4 are user distance classes UD50-UD80, 5,6,7,8 are tablet pose classes neutral, roll, pitch, yaw.

For feature selection, the relative importance of features (Eqn.15) are estimated using a random forest classifier method [33]–[35]. This model comprises of a number of decision trees in which it can be computed how much each feature reduces the weighted impurity [36]in the tree. The total impurity decrease contributed by each feature is averaged to rank the feature's importance. Features computed on desktop and tablet datasets are fed to a random tree classifier model with hyper-parameters (n_estimators=200, max_depth=8). The rankings of features for desktop and

tablet datasets are shown in Fig.14 a and b. In all the datasets, mean, standard deviation, IQR and confidence intervals (feature numbers 16-20) emerge as the most significant features. Based on this, the following reduced feature set was computed: [μ, σ, 95%Conf_up, 95%Conf_down, IQR]. This reduced feature set was used with SVM and KNN for classification. However neural network models required the full feature set to reduce training error and prevent under-fitting.

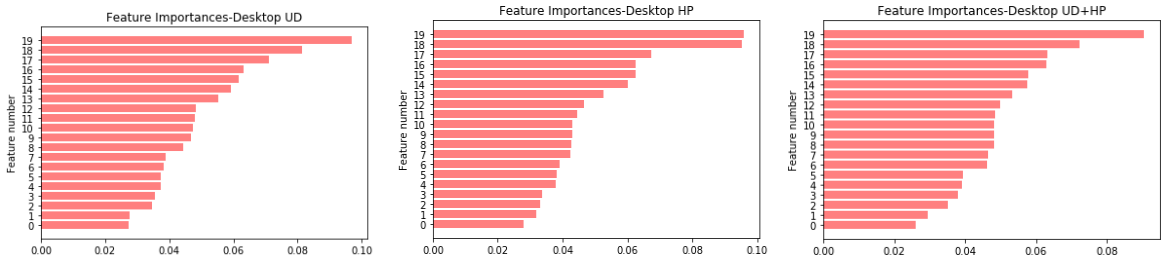


Fig 14a: Relative importance of features for different desktop datasets

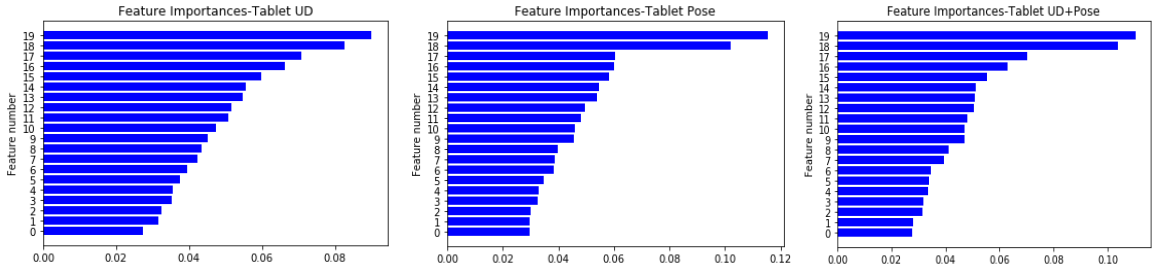


Fig 14b: Relative importance of features for different tablet datasets

Table 3. Training and test dataset details

Samples for train, test	**Original feature set (20 features)**	**Augmentation strategies**	**Samples per person**	**Samples and classes**
Total subjects: 20 12- 16 subjects for training, 8-5 for test Data labelled and randomly shuffled	1) Gaze error values at 15 AOIs, Mean, SD, IQR, 0.95 interval bounds of gaze error 2) Yaw error values at 15 AOIs, Mean, SD, IQR, 0.95 interval bounds of yaw error 3) Pitch error values at 15 AOIs, Mean, SD, IQR, 0.95 interval limits of pitch error **Reduced feature set (5 features)** Mean, SD, IQR, 0.95 interval bounds for each sample.	1) Gaussian noise 2) Jitter or pink noise 3)Horizontal data flipping 4) Vertical data flipping 5) Magnitude warping 6)Time warp 7) Interpolation & combinations	10 x gaze error 10 x yaw error 10 x pitch error **Merged dataset** 30 samples	1) Desktop user dist, 2400 samples, 4 classes 2) Head pose, 2400 samples, 4 classes 3) Desktop mixed: 4200 samples, 7 classes 5) Tablet user dist, 2400 samples, 4 classes 6) Tablet pose, 2400 samples, 4 classes 7) Tablet mixed: 4200 samples, 7 classes

3.4 Classification models: k-NN, SVM and ANN

After data augmentation and exploration of the feature set and labelling, the datasets were used with three different machine learning models from the Python Scikit-Learn libraries, run on a Windows 7 computer with core i7 2.6 GHz processor. A brief description of the models is below.

a) K-nearest neighbours (k-NN): In this, the underlying assumption is that samples of the same class will be the nearest neighbors of each other i.e. the distance between them will be small since they are related[37][38]. While using this method, the training samples form vectors in a multidimensional feature space, each with its class label. In the training phase of the algorithm the feature vectors are stored along with class labels of the training samples. In the classification phase, the distance between an un-labelled input vector and the training dataset is calculated to get the k (a user defined hyper-parameter number) nearest points of the input sample. Then the input sample is categorized into the class of the majority in the k nearest points. Thus if the training data is $\{(x_1,y_1),(x_2,y_2)\ldots(x_N,y_N)\}$, and x is the feature vector of an input sample, the KNN finds the class of the input sample x using a distance metric [39][40] which in this work is the Euclidean distance given by:

$$\text{Dist}(x_i,x_j) = \sqrt{\left(\sum_{i=1}^{N}(x_i - x_j)\right)^2} \qquad (17)$$

b)Support Vector Machines(SVM): It is a supervised learning method [41][42]that works on the principle of transforming the input feature space by a nonlinear transformation to a high dimensional feature space, and searching for an optimal separating hyperplane for the input classes in this new high-dimensional space[43][44]. With this separating hyperplane the training data x_i with labels y_i can be classified so that the minimal distance of each point from the hyperplane is maximized. With the training data depicted as instance-label pair (x_i, y_i), i=1,...,m

where $x_i \in R^n$ represents the input vector and $y_i \in (-1,1)$ are the corresponding output label of x_i. The objective function for SVMs can be defined as:

$$\min_{w, b, \xi} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{N} \xi^2 \quad (18)$$

with the condition: $y_i(w^T x + b) \geq 1 - \xi_i^2$ (i=1,…N). (19)

Here C >0 is the regularization parameter and ξ_i is the slack variable, w is the weight vector and b is the offset. To classify un-labelled examples x_k according to labels y_k using Kernel function $K(x_i, x_k)$, the optimal separating hyperplane equation becomes:

$$y_k = \text{sign}(\sum_{x_i \epsilon S} a_i y_i K(x_i, x_k) + b) \quad (20)$$

where S is the set of support vectors xi, ai are Lagrange multipliers (used for solving the optimization problem). In this work, a Gaussian Radial Basis Function or RBF kernel is used which is given by:

$$K(x_i, x_k) = \exp[-\gamma \|x_i - x_k\|^2] \quad (21)$$

The hyper-parameters C and γ are chosen for optimal fitting and discussed in next section.

c) Multilayer neural networks (MLP): There are supervised learning algorithms [45]–[48] which takes in labelled training examples and solves complex, non-linear hypothesis by a network of computing units called "neurons". Learning in an MLP takes place by updating connection weights of each neuron after passing a batch of data samples from input to output, depending on the amount of error in the output compared to the target. The general update rule for weights (Δw_{ji}) in an MLP based on backpropagation and gradient descent is:

$$\Delta w_{ji} = - \eta \frac{\partial E(n)}{\partial w_{ij}} y_i(n) \quad (22)$$

Where, $y_i(n)$is the output of the previous neuron and η is the learning rate, E is the error in the nth node. In this work, neural networks with ReLU activation, Adam optimizer and constant learning rate of 0.001 is used.

3.5 Classification results on desltop data

3.5.1 Results, Task I: Classification of desktop data for different user distances

The KNN, SVM and MLP models were used for classification of gaze datasets corresponding to four different classes of user distances (50, 60, 70, 80 cm) and the results are in Table 4. In the table, Tr, Ts, Cv are training, test and cross validation accuracies. A training-test sample proportion of 30% and 10-fold cross validation was used in all cases[49]. For all models, grid search [50]with cross validation is used to determine the optimal hyper-parameters. For K-NN, the optimal number of neighbors was found to be 3. For SVM, the RBF kernel is used [51]with the soft margin cost function parameter C set to 10 and gamma set to 1.0. C defines the trade-off between misclassification and simplicity of the decision surface. Gamma is the RBF kernel parameter (Eqn. 21). For MLP, grid-search yielded a best hidden layer size of 3 with neuron configurations of [50, 100, 50] units in the layers with full training set of 20 features being used.

3.5.2 Results, Task II: Classification of desktop data for different head poses

In this task, the gaze datasets collected from desktop setup for different user head poses are used. The head pose datasets include gaze error values for head roll (20 degrees), pitch(20 degrees), yaw(20 degrees) and neutral (roll, pitch, yaw= 0 degrees). KNN, SVM and MLP models were used and classification accuracies for the different classifiers are tabulated in Table 5.

Using grid search, for KNN, the best number of neighbors was found to be 3 and reduced feature set was used. For SVM, a C value of 10 and gamma of 1.25 was used with the reduced feature set. For the MLP model, a 3 layer network was used and its architecture is tuned by varying the number of units in each layer between (50-100) and varying the regularization parameter values between 0.001 to 0.5 to control overfitting. Also the full feature set was used for training.

3.5.3 Results, Task III: Classification on merged user distance and head pose datasets from desktop

In this task, the head-pose and user distance datasets are merged and classifiers are applied to do a 7 class classification on this mixed dataset. Since the head pose data was collected at a user distance of 60 cm, the neutral head pose data and the UD60 data are the same, and therefore only one of these two datasets is used to avoid class imbalance in the mixed dataset. As above, the three classifiers are trained and tested on this dataset. For the KNN, the best number of neighbors was chosen to be 3 as shown in Fig 16c below, which shows the dependence of train, test and cross validation error as a function of number of neighbors used in the model. For the SVM model, the parameters were same for the above two tasks. For the MLP, 2 hidden layers with 100 units each and regularization value of 0.001 was used with the full set of features. Classification results are presented in Table 6 and confusion matrix for MLP model is in Fig 15. Table 7 shows the performance of the three classifiers on different datasets, with true and false detection rates and precision values. It is observed that for desktop data, KNN and MLP classification performances are close, with the KNN performing well for all the datasets.

Table 4: Classifier performance for user distance

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.91 Ts= 0.9 Cv= 0.89	Tr= 0.73 Ts= 0.70 Cv=0.69	Tr=0.95 Ts=0.81 Cv=0. 86
Only gaze yaw features	Tr=0.90 Ts=0.83 Cv= 0.83	Tr= 0.73 Ts= 0.64 Cv= 0.64	Tr=0.98 Ts=0.87 Cv= 0.84
Only gaze pitch features	Tr=0.98 Ts= 0.95 Cv=0.96	Tr=0.89 Ts= 0.90 Cv=0.89	Tr=0.96 Ts=0.82 Cv=0.73
Merged (gaze, yaw, pitch) feature dataset	Tr= 0.90 Ts=0.75 Cv= 0.84	Tr= 0.70 Ts=0.64 Cv= 0.67	Tr=0.92 Ts=0.78 Cv=0.81

Table 5: Classifier performance for head pose

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.95 Ts= 0.88 Cv= 0.91	Tr= 0.86 Ts= 0.80 Cv=0.83	Tr=0.98 Ts=0.83 Cv=0.90
Only gaze yaw features	Tr=0.92 Ts=0.84 Cv= 0.88	Tr= 0.87 Ts= 0.77 Cv= 0.78	Tr=0.98 Ts=0.87 Cv= 0.86
Only gaze pitch features	Tr=0.98 Ts= 0.97 Cv=0.95	Tr=0.98 Ts= 0.96 Cv=0.96	Tr=0.96 Ts=0.85 Cv=0.83
Merged (gaze, yaw, pitch) feature dataset	Tr= 0.93 Ts=0.85 Cv= 0.89	Tr= 0.86 Ts=0.79 Cv= 0.80	Tr=0.92 Ts=0.82 Cv=0.84

Table 6: Classification on mixed datasets (desktop)

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.91 Ts= 0.80 Cv= 0.87	Tr= 0.80 Ts= 0.74 Cv=0.74	Tr=0.94 Ts=0.76 Cv=0.83
Only gaze yaw features	Tr=0.88 Ts=0.73 Cv= 0.81	Tr= 0.72 Ts= 0.65 Cv= 0.66	Tr=0.97 Ts=0.82 Cv= 0.80
Only gaze pitch features	Tr=0.97 Ts= 0.91 Cv=0.92	Tr=0.93 Ts= 0.91 Cv=0.91	Tr=0.98 Ts=0.83 Cv=0.79
Merged (gaze fontal, yaw, pitch) feature dataset	Tr= 0.91 Ts=0.78 Cv= 0.85	Tr= 0.77 Ts=0.70 Cv= 0.70	Tr=0.92 Ts=0.75 Cv=0.77

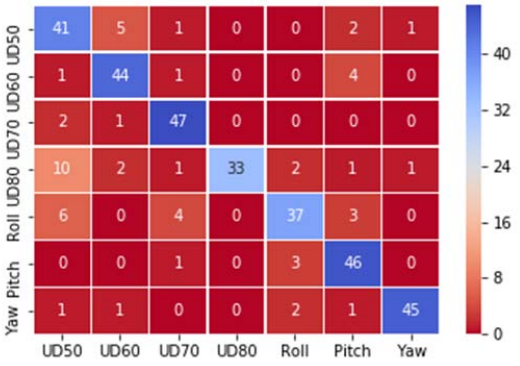
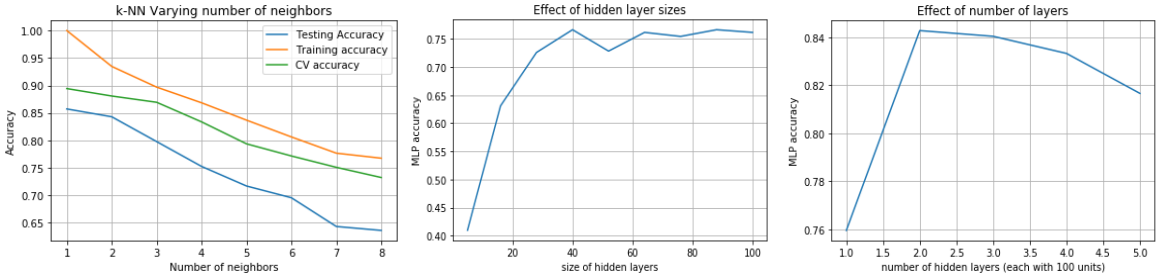


Fig 15. Confusion matrix using MLP on mixed head pose and user distance dataset



(a) (b) (c)

Fig.16 Effect of varying model hyper-parameters for KNN (a) and MLP(b, c)

Table 7: True and false detection rates from classifiers- Desktop (for user distance, head pose and mixed datasets)

User distance dataset				**Head pose dataset**				**Mixed head pose, user distance dataset**			
	KNN	**SVM**	**MLP**		**KNN**	**SVM**	**MLP**		**KNN**	**SVM**	**MLP**
TPR	0.96	0.95	0.85	**TPR**	0.97	0.91	0.95	**TPR**	0.83	0.93	0.83
FPR	0.01	0.01	0.04	**FPR**	0.01	0.02	0.01	**FPR**	0.02	0.01	0.02
TNR	0.98	0.98	0.95	**TNR**	0.99	0.97	0.98	**TNR**	0.97	0.98	0.97
FNR	0.04	0.04	0.14	**FNR**	0.03	0.08	0.04	**FNR**	0.16	0.06	0.16
Precision	0.98	0.93	0.85	**Precision**	0.97	0.97	0.95	**Precision**	0.85	0.93	0.85

3.6 Classification results on tablet data

In this subsection, results from using tablet data with various classification models are presented. These include datasets for different user-tablet distances (50, 60, 70, 80 cm) and tablet poses of neutral, platform roll, pitch and yaw of 20 degrees. A mixed dataset is created by merging the datasets for user distance and platform pose and classifier models are trained to do seven class classifications i.e. distinguish between 4 user distance and 3 tablet pose classes.

3.6.1 Results, Task IV: Classification of tablet data for different user distances

For the user distance dataset from tablet platform, the KNN classifier with 3 neighbors is used. The SVM is used with RBF kernel, C= 10 and gamma of 1 and the MLP classifier is used with 3 layers, 200 units each and with a regularization value of 0.0001. The best performance is by MLP and classification results are in Table 8.

3.6.2 Results, Task V: Classification of tablet data for different tablet poses

For the tablet pose dataset, the KNN classifier with 3 neighbors are used. The SVM is used with RBF kernel, C= 5 and gamma of 0.5. The MLP model is used with 2 layers, 100 units each and a regularization value of 0.0001. Best classification results are again by MLP and are in Table 9.

3.6.3 Results, Task VI: Classification of tablet data for mixed user distance and tablet pose datasets

For the mixed user distance and platform pose datasets, the KNN classifier with 3 neighbors is used. The SVM is used with RBF kernel, C= 5 and gamma of 0.5 and the MLP classifier is used with 2 layers, 100 units each and with a regularization value of 0.0001. Both SVM and MLP perform well and classification results are in Table 10.

Table 8: Classifier accuracy on tablet distance datasets

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.89 Ts= 0.78 Cv= 0.81	Tr= 0.94 Ts= 0.78 Cv= 0.82	Tr= 0.95 Ts= 0.77 Cv=0.81
Only gaze yaw features	Tr=0.91 Ts=0.83 Cv=0.84	Tr= 0.98 Ts= 0.82 Cv= 0.86	Tr= 0.96 Ts= 0.78 Cv= 0.78
Only gaze pitch features	Tr= 0.94 Ts= 0.90 Cv=0.90	Tr= 0.94 Ts= 0.78 Cv=0.81	Tr= 0.78 Ts= 0.70 Cv= 0.71
Merged (gaze fontal, yaw, pitch) feature dataset	Tr= 0.90 Ts=0.8 Cv=0.85	Tr= 0.97 Ts= 0.78 Cv= 0.84	Tr= 0.91 Ts= 0.74 Cv= 0.76

Table 9: Classification on tablet pose datasets

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.91 Ts= 0.79 Cv= 0.86	Tr= 0.99 Ts= 0.82 Cv= 0.88	Tr= 0.99 Ts= 0.75 Cv= 0.85
Only gaze yaw features	Tr= 0.96 Ts= 0.87 Cv= 0.93	Tr= 0.99 Ts= 0.92 Cv= 0.92	Tr= 0.98 Ts= 0.85 Cv= 0.90
Only gaze pitch features	Tr= 0.96 Ts= 0.91 Cv= 0.94	Tr= 0.99 Ts= 0.9 Cv= 0.91	Tr=0.82 Ts= 0.75 Cv= 0.78
Merged (gaze fontal, yaw, pitch) feature dataset	Tr= 0.92 Ts= 0.86 Cv= 0.88	Tr= 0.98 Ts= 0.85 Cv= 0.85	Tr= 0.89 Ts= 0.74 Cv= 0.84

Table 10: Classification on tablet(distance+ pose) dataset

Dataset used	k-NN	SVM	Neural network
Only gaze angle features	Tr= 0.88 Ts= 0.73 Cv=0.75	Tr= 0.95 Ts= 0.79 Cv= 0.80	Tr= 0.96 Ts= 0.73 Cv= 0.78
Only gaze yaw features	Tr= 0.95 Ts=0.82 Cv= 0.85	Tr= 0.96 Ts= 0.78 Cv= 0.84	Tr= 0.88 Ts= 0.73 Cv= 0.70
Only gaze pitch features	Tr= 0.92 Ts= 0.82 Cv= 0.89	Tr= 0.95 Ts= 0.82 Cv= 0.84	Tr= 0.86 Ts= 0.72 Cv= 0.79
Merged (gaze fontal, yaw, pitch) feature dataset	Tr= 0.90 Ts= 0.76 Cv= 0.82	Tr= 0.98 Ts= 0.79 Cv= 0.83	Tr= 0.84 Ts= 0.67 Cv= 0.75

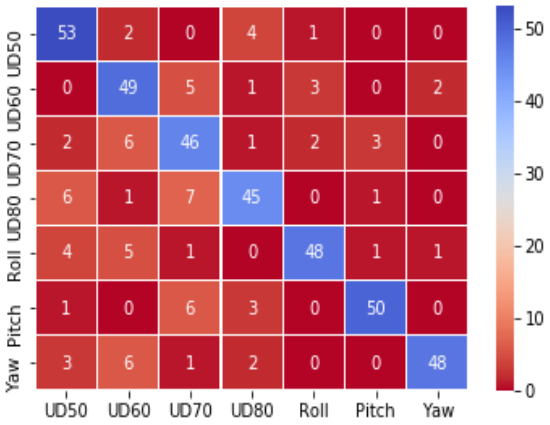


Fig. 17 Confusion matrix using MLP on mixed tablet (user distance +tablet pose) dataset

Table 11: True and false detection results from classifiers-Tablet (for user distance, tablet pose and mixed datasets)

User distance dataset				**Platform pose dataset**				**Mixed user distance, tablet pose dataset**			
User-Dist	**KNN**	**SVM**	**MLP**	**Pose**	**KNN**	**SVM**	**MLP**	**Mixed**	**KNN**	**SVM**	**MLP**
TPR	0.87	0.85	0.71	**TPR**	0.90	0.83	0.79	**TPR**	0.80	0.83	0.70
FPR	0.04	0.04	0.09	**FPR**	0.03	0.27	0.06	**FPR**	0.03	0.02	0.04
TNR	0.95	0.95	0.90	**TNR**	0.96	0.97	0.93	**TNR**	0.96	0.97	0.95
FNR	0.12	0.14	0.28	**FNR**	0.09	0.16	0.20	**FNR**	0.19	0.16	0.29
Precision	0.87	0.88	0.72	**Precision**	0.91	0.86	0.79	**Precision**	0.82	0.88	0.72

3.7 Discussions

The results from this section demonstrate that ML models can distinguish between gaze data collected under normal and varying operating conditions. Thus using these models, anomalies present in gaze datasets may be detected. This is similar to anomaly detection approaches [52]–[54] used in fields such as cyber intrusion and video surveillance, where training sets of normal and nominal examples are used to design a decision rule such that occurrences of erroneous data samples are detected accurately.

For the classification tasks, gaze datasets were constructed such that they contained signatures of a single or multiple error sources. It was found through the decision tree based feature selection

technique that statistical attributes such as gaze error confidence levels and interquartile ranges are significant parameters that can be used to distinguish gaze error sources. In the classification tasks, the KNNs and MLP classifiers performed best for all datasets, although MLP models take more time to converge. Also, it is seen from Tables 4-6 and 8-10 that using gaze yaw and pitch features result in better detection accuracies than gaze angular datasets. From Tables 7 and 11, it is seen that the rate of false detections is also low for all the models. All the datasets, especially the ones created with 7 classes, were found to be quite complex as no class separation could be detected from t-SNE results. Despite this, the classification models achieved a cross validation score of 85-90% on most datasets. This shows the feasibility of ML models in detecting gaze error sources even from complex gaze datasets where more than one source is present.

The ML models used here were optimized through experimentation to select the set of hyper-parameters that best improve their cross validation scores. In some cases, results show under-fitting, especially in 7 class classification problems, which are inherently complex. For this kind of problems with little separation between classes, larger training datasets can improve results.

4 Modelling and prediction of gaze errors

Using the collected desktop and tablet datasets, regression models [55][56]are trained to predict gaze estimation errors using gaze angle, yaw and pitch angle values as input features. Two different models are built which may be used to predict gaze errors of an eye tracker under two different operating conditions. These include the head-pose error model, which can be used to predict gaze errors occurring due to various user head poses. The other is the platform pose error model, for predicting gaze error corresponding to different poses of the eye tracker. The gaze errors are predicted as a function of a user's gaze angle and gaze yaw/pitch angles (Eqn 28). The results from the various gaze error models are presented in sub-section 4.1 and 4.2 below.

A regression model describes a dependent variable Y as a function of an independent variable x with the generic relation: Y = a + b1X1 + b2X2 + b3X3 + ... + btXt + u, where X1, X2.. Xt are the independent variables, a is the intercept, b is the slope and u is the regression residual[57]. The cost function used for evaluation of model fit is given by root mean squared error or RMSE:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \quad (23)$$

where n is the number of observations, yi is the true value of target to predict, and $\hat{y}_i$ is model's predicted result. The optimization function of a standard linear regression can be expressed as:

$$\min \| Xw - y \|^2 \quad (24)$$

where X is the set of feature variables, w represents the weights, and y comprises of the ground truth data points. In this work, several regression models are used, including Ridge, Lasso, ElasticNet [58][59] and Neural network based models[60]. In Ridge regression the issue of high variance is mitigated through addition of a squared bias factor as regularization in the form:

$$\min \| Xw—y \|^2 + z\| w \|^2 \quad (25)$$

Whereas in Lasso regression, an absolute value bias of the form below is used:

$$\min \| Xw—y \|^2 + z\| w \| \quad (26)$$

The ElasticNet regression uses the regularization factors of both the above techniques:

$$\min \| Xw—y \|^2 + z\_1\| w \| + z\_2\| w \|^2 \quad (27)$$

In this work, the regression models are used to map gaze frontal and yaw, pitch angle values to gaze errors produced under different operating conditions. The task of the regression algorithms is not only to find the mapping between input and output variables but also take into consideration the interactions between the input variables. The input features (X1, X2, X3) are:

$$[\text{Gaze\_Angle, Gaze\_Yaw, Gaze\_Pitch}] \rightarrow \text{Gaze error} \quad (28)$$

For the regression tasks, the input features are standardized such that their distributions have mean value of 0 and standard deviation of 1. Six different regression models for predicting gaze errors are trained on the features created using Eqn. 28 on head pose(desktop) and platform pose

(tablet) datasets. For each model, fit estimates in RMSE values are summarized in Tables 12 (Desktop) and 13 (tablet). The coefficients of the best performing models are listed in Table 14.

4.1 Head pose error model

Table 12: Prediction errors (RMSE) for models-Desktop

	Dataset: Head pose-Desktop			
Model	**Neutral**	**R20**	**P20**	**Y20**
Linear	2.24	1.36	1.45	2.71
Polynomial	7.48	1.82	7.88	2.88
Ridge	2.24	1.36	1.44	2.70
Lasso	2.24	1.15	1.31	2.70
ElasticNet	2.25	1.07	1.29	2.69
Neural network	4.01	1.21	2.09	2.75

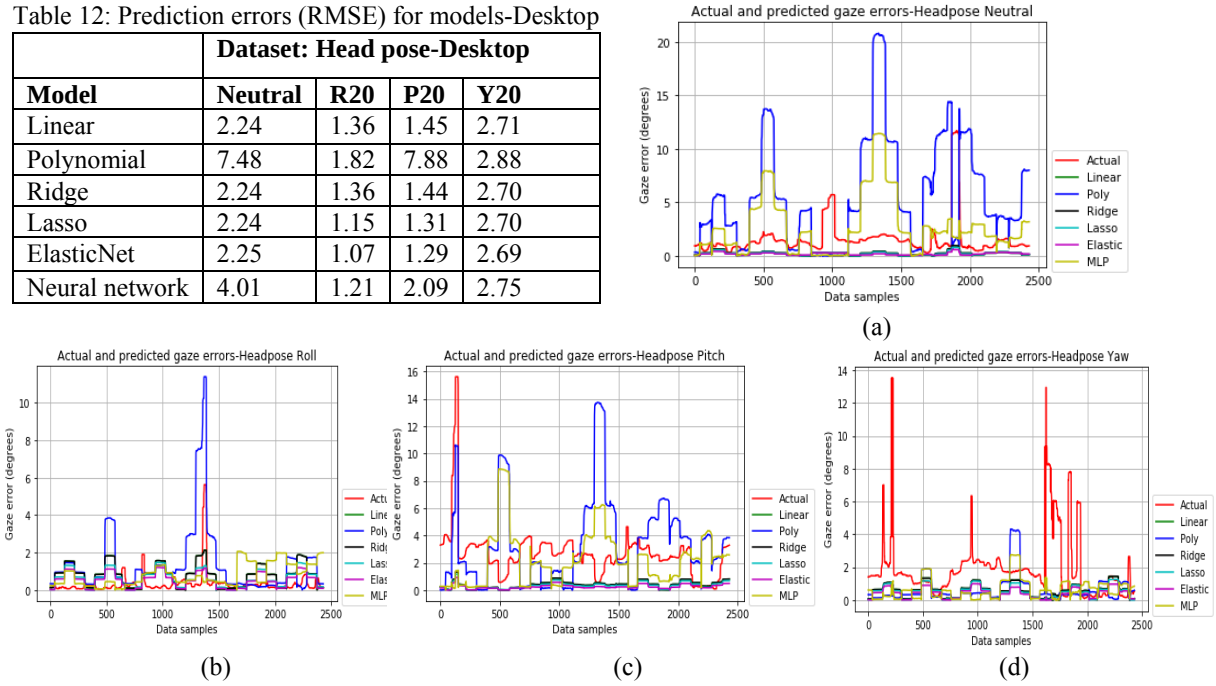


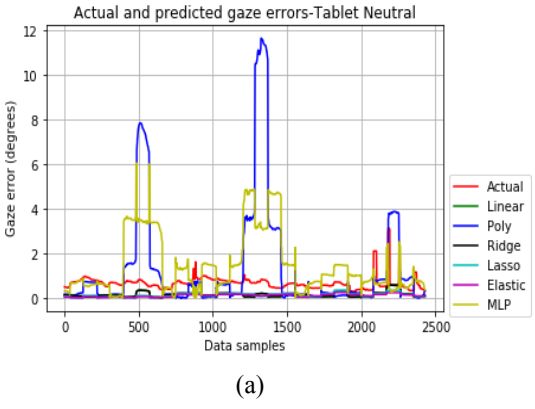
Fig. 18 Actual gaze errors (red) and predicted gaze errors for head pose neutral, roll, pitch, yaw (a-d) datasets

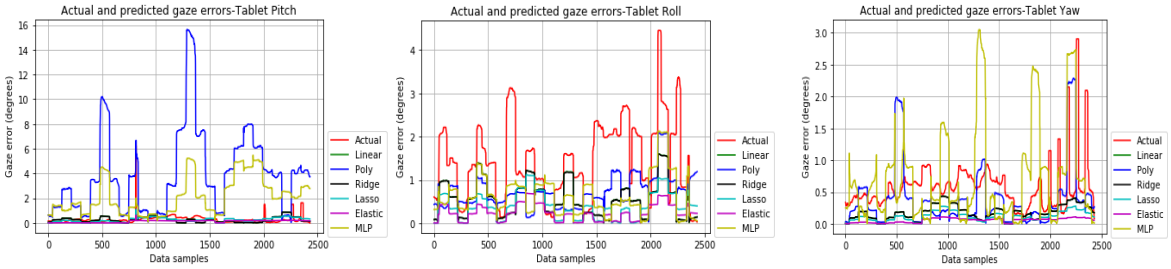
Table 12 shows that ElasticNet (with penalty parameter =0.5) has the lowest prediction errors for all head pose datasets. Regularization used for Ridge and Lasso models is 0.001. MLP model is used with 1 hidden layer with 100 units, ReLU activation and regularization value of 0.001.

4.2 Platform pose error model

Table 13: Prediction errors (RMSE) for models-Tablet

	Dataset: Platform pose-Tablet			
Model	**Neutral**	**R20**	**P20**	**Y20**
Linear	0.73	1.79	0.58	0.72
Polynomial	2.61	1.99	5.24	0.81
Ridge	0.73	1.79	0.58	0.72
Lasso	0.73	1.73	0.58	0.71
ElasticNet	0.73	1.71	0.50	0.70
Neural network	1.75	1.78	2.39	1.18





(b) (c) (d)

Fig. 19 Actual gaze errors (red) and predicted gaze errors for tablet pose neutral, roll, pitch, yaw (a-d) datasets

For the tablet pose dataset, the ElasticNet performs well, with a penalty parameter value of 0.5. The MLP and polynomial regression models are seen to overestimate the error values. It was seen that outliers and noise strongly affect all the models and therefore outlier removal and data standardization methods to convert the input features to have normal distribution is essential.

4.3 Establishment of the error models

As described above, a regression model with three predictor variables can be expressed as:

$$Y = B_0 + B_1*X_1 + B_2*X_2 + B_3*X_3 \quad (28)$$

Where B_0 is the intercept, B_1, B_2 and B_3 are coefficients and X_1, X_2, X_3 are input features (gaze, yaw, pitch angles). Since the ElasticNet model had lowest RMSE for predicting head and tablet pose errors, the B_0, B_1, B_2 and B_3 parameters of this model are computed for the head pose and tablet pose datasets and presented in Table 14. With these parameters, the head and platform pose error models may be constructed for error prediction using gaze angular variables as input.

Table 14: Coefficients and intercept of the best model for different datasets

Platform/Condition	**ElasticNet model coefficients: B_1, B_2 and B_3**	**Intercept B_0**
Desktop: Head pose neutral	[0.09336917, 0.19406989, -0.00279198]	-1.99912371e-16
Desktop: Head Roll 20	[0. , 0.65189252, 0.07303053]	3.23942926e-16
Desktop: Head Pitch 20	[0.22606558, 0.11028886, 0.05731872]	-9.55229176e-17
Desktop: Head Yaw 20	[0. , 0.51352565, 0.08149052]	8.94037155e-17
Tablet: Platform pose neutral	[0.07333954, 0. , -0.17956056]	1.76076146e-16
Tablet: Platform Roll 20	[0. , -0.31460996, -0.23620848]	-2.87637414e-16
Tablet: Platform Pitch 20	[0. , -0.05682588, -0.20804325]	2.34007877e-16
Tablet: Platform Yaw20	[0. , -0.01596391, -0.06346607]	-2.41027682e-17

5 Conclusion and future work

In contemporary eye gaze research, there is scarcity of analytical methods for studying the variability of gaze accuracy in eye tracking data acquired under unconstrained conditions. There is also lack of datasets that provide gaze and ground truth information for different operating conditions, and gaze datasets labelled with different experimental scenarios are absent. Therefore, in this work, a new and diverse gaze dataset is collected specifically for this purpose, and labelled with different operating conditions. The dataset comprises of fixation data from 20 participants captured from different user platforms and operating conditions such as 6 different head poses, 4 different user distances and 6 platform poses. This dataset is then analyzed to identify the presence and modelling the impacts of the above conditions on gaze error levels estimated from the collected data. The details of the dataset are in the link below and the gaze data files are available from the authors on request. It will also be publicly available shortly.

Link: https://data.mendeley.com/datasets/2txnw3y6gs/draft?a=6ca3629f-393c-4f80-93f6-8d56e6557792 .

Several new approaches with respect to gaze error pattern analysis have been implemented in this work. This includes the use of several strategies for outlier removal, and the median filtering approach among others was highly successful in de-noising gaze data from all the collected datasets. Next, multiple gaze data augmentation methods were applied which helped to increase the size of the collected gaze datasets by an order of magnitude and introduce lots of variabilities within the datasets. The choice of gaze error features and use of random forest based feature selection method helped to reveal insights about gaze error characteristics, e.g. that interquartile range and confidence intervals are significant indicators for distinguishing gaze error patterns.

The t-SNE algorithm is a relatively new concept in machine learning which was used in this work to visualize the distribution of classes within the gaze datasets, when gaze data influenced

by different error sources are mixed together. This technique demonstrated the complexities of the gaze datasets used in this work, which are close to what might be expected in gaze data from un-constrained practical applications. Finally, the use of machine learning for classification and prediction of gaze error patterns in artificially created heterogeneous gaze datasets as done in this work is a new concept, that has not been explored before.

This work also reveals that ML models could be highly robust in identifying gaze error patterns even in complex gaze datasets. It was seen that classifier models can successfully distinguish between eye tracking data collected under normal conditions (corresponding to neutral head and platform poses), and data collected under varying operating conditions such as high degree of head pose variations, various user distances and platform poses. With these classifiers, known types of error sources in eye gaze datasets may be detected, and gaze error patterns that do not match the normal gaze behavior may be identified and recognized as new error types.

The various concepts developed in this work, including application of classifier and regression models to gaze error data may be used for building improved eye tracking systems and algorithms and also getting better results from them. The main benefit of training the classifier models is that they can distinguish anomalous gaze data present in realistic and complex gaze datasets as used in this work and possibly recognize what caused them (e.g. impact of user distance or head pose). This can help eye gaze application developers, researchers or engineers to deploy appropriate prevention methods, compensation strategies or setup improvements to reduce the impacts of the error sources affecting their data.

The regression models may be used predict how an eye tracker might behave under practical operating conditions like high degree of head pose or platform pose variations and forecast the

possible levels of error caused by different error sources. This can help the eye gaze researchers or engineers to quantitatively specify the limits of their systems while operating under these challenging conditions and also develop suitable error correction methods.

Also the outlier detection techniques described in this work could be useful for applying to any gaze dataset prior to analyzing them, to observe underlying patterns. Overall, the gaze data analysis pipeline as implemented in this work could be of use in parts or whole to any eye gaze researcher or engineer who wishes to gain deeper insights into their collected gaze data and have an estimate about the impacts of various non-ideal operating conditions.

There also remains several ways to extend the work presented here since extracting and analyzing error patterns from gaze data is a new and unexplored field in gaze research. For example, deep neural networks may be used for detecting anomalous gaze data collected from unconstrained setups. As mentioned in Section 1.2, in this work, influence of one operating is considered at a time while collecting gaze data. However, complex scenarios may arise where gaze data may be affected by more than one error source, as for example in automotive use cases. The problem of multi-factor influence on gaze data is an interesting and complex research question but can only be answered when relevant and labelled data is available. Therefore collecting relevant gaze datasets from complex operational scenarios using desktop, handheld, head mounted or automotive setups towards understanding diverse range of gaze error patterns could be another potential future work in this domain.

Acknowledgement: The research work presented here is funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded by FotoNation Ltd. Project ID: 13/SPP/I2868 on “Next Generation Imaging for Smartphone and Embedded Platforms”.

References

1. Anuradha Kar, Peter Corcoran: Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations. Sensors 18(9): 3151 (2018)..
2. M. Cheon and J. Lee, "Gaze pattern analysis for video contents with different frame rates," *2013 Visual Communications and Image Processing (VCIP)*, Kuching, 2013, pp. 1-5.
3. Zhu D., Mendis B.S.U., Gedeon T., Asthana A., Goecke R. (2009) A Hybrid Fuzzy Approach for Human Eye Gaze Pattern Recognition. In: Köppen M., Kasabov N., Coghill G. (eds) Advances in Neuro-Information Processing. ICONIP 2008. Lecture Notes in Computer Science, vol 5507. Springer, Berlin, Heidelberg.
4. Y. Horiguchi, T. Suzuki, T. Sawaragi, H. Nakanishi and T. Takimoto, "Extraction and Investigation of Dominant Eye-Gaze Pattern in Train Driver's Visual Behavior Using Markov Cluster Algorithm," *2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS)*, Sapporo, 2016, pp. 578-581..
5. Christian Braunagel, David Geisler, Wolfgang Stolzmann, Wolfgang Rosenstiel, and Enkelejda Kasneci. 2016. On the necessity of adaptive eye movement classification in conditionally automated driving scenarios. In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16). ACM, New York, NY, USA, 19-26..
6. F. Koochaki and L. Najafizadeh, "Predicting Intention Through Eye Gaze Patterns," *2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Cleveland, OH, 2018, pp. 1-4.
7. [Pekkanen, J., Lappi, O.: A new and general approach to signal denoising and eye movement classification based on segmented linear regression. Scientific Reports 7(1), 1–13 (2017).
8. Zemblys, R., Niehorster, D.C., Komogortsev, O. et al. Behav Res (2018) 50: 160.
9. Michael Barz, Florian Daiber, Daniel Sonntag, and Andreas Bulling. 2018. Error-aware gaze-based interfaces for robust mobile gaze interaction. In Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18). ACM, New York, NY, USA, Article 24, 10 pages. .

10. Michael Barz, Florian Daiber, and Andreas Bulling. 2016. Prediction of gaze estimation error for error-aware gaze-based interfaces. In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16). ACM, New York, NY, USA, 275-278.
11. [T. Ishikawa, S. Baker, I. Matthews, and T. Kanade, “Passive Driver Gaze Tracking with Active Appearance Models,” Proc. World Congr. Intell. Transp. Syst., pp. 1–12, 2004.
12. Tao Chen, Kai-Kuang Ma and Li-Hui Chen, "Tri-state median filter for image denoising," in IEEE Transactions on Image Processing, vol. 8, no. 12, pp. 1834-1838, Dec. 1999..
13. H. H. Khalil, R. O. K. Rahmat and W. A. Mahmoud, "Chapter 15: Estimation of Noise in Gray-Scale and Colored Images Using Median Absolute Deviation (MAD)," 2008 3rd Intl. Conference on Geometric Modeling and Imaging, London, 2008, pp. 92-97..
14. Y.C. Chen, “A tutorial on kernel density estimation and recent advances,” Biostat. Epidemiol. 1, 161–187 (2017). .
15. [Hatice Ceylan Koydemir, Steve Feng, Kyle Liang, Rohan Nadkarni, Derek Tseng, Parul Benien, Aydogan Ozcan, "A survey of supervised machine learning models for mobile-phone based pathogen identification and classification," Proc. SPIE 10055, Optics and Biophotonics in Low-Resource Settings III, 100550A (7 March 2017).
16. Qiu, J., Wu, Q., Ding, G. et al. EURASIP J. Adv. Signal Process. (2016) 2016: 67.
17. [Bjerrum, Esben Jannik, Mads Glahder and Thomas Skov. “Data Augmentation of Spectral Data for Convolutional Neural Network (CNN) Based Deep Chemometrics.” CoRR abs/1710.01927 (2017).
18. Polson, Nicholas G.; Scott, Steven L. Data augmentation for support vector machines. Bayesian Anal. 6 (2011), no. 1, 1--23. doi:10.1214/11-BA601. https://projecteuclid.org/euclid.ba/1339611936..
19. M.J. Sáiz-Abajo, B.-H. Mevik, V.H. Segtnan, T. Næs, Ensemble methods and data augmentation by noise addition applied to the analysis of spectroscopic data, Analytica Chimica Acta, Volume 533, Issue 2, 2005, Pages 147-159, ISSN 0003-2670.

20. Andrew T. Duchowski, Sophie Jörg, Tyler N. Allen, Ioannis Giannopoulos, and Krzysztof Krejtz. 2016. Eye movement synthesis. In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16). ACM, New York, NY, USA, 147-154.
21. Andrew Duchowski, Sophie Jörg, Aubrey Lawson, Takumi Bolte, Lech Świrski, and Krzysztof Krejtz. 2015. Eye movement synthesis with 1/f pink noise. In Proc. 8th ACM SIGGRAPH Conf.on Motion in Games (MIG '15). ACM, New York, USA, 47-56.
22. T. Devries and G. W. Taylor, “Dataset Augmentation in Feature Space", arXiv:1702.05538v1 [stat.ML], pp. 1–12, 2017..
23. Terry T. Um, Franz M. J. Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulić. 2017. Data augmentation of wearable sensor data for parkinson’s disease monitoring using convolutional neural networks. In Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI '17). ACM, New York, NY, USA, 216-220.
24. J. Salamon and J. P. Bello, "Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification," in IEEE Signal Processing Letters, vol. 24, no. 3, pp. 279-283, March 2017.
25. L. Li, Y. Wu, Y. Ou, Q. Li, Y. Zhou and D. Chen, "Research on machine learning algorithms and feature extraction for time series," 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, 2017, pp. 1-5.
26. M. C. Popescu and L. M. Sasu, "Feature extraction, feature selection and machine learning for image classification: A case study," 2014 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), Bran, 2014, pp. 968-973.
27. S. Khalid, T. Khalil and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," 2014 Science and Information Conference, London, 2014, pp. 372-378..
28. M. Oravec, "Feature extraction and classification by machine learning methods for biometric recognition of face and iris," Proceedings ELMAR-2014, Zadar, 2014, pp. 1-4.

29. L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9(Nov):2579-2605, 2008.
30. [N. Rogovschi, J. Kitazono, N. Grozavu, T. Omori and S. Ozawa, "t-Distributed stochastic neighbor embedding spectral clustering," 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 1628-1632.
31. S. Mounce, “Visualizing Smart Water Meter Dataset Clustering With Parametric T-distributed Stochastic Neighbour Embedding,” 2017 13th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov, pp. 1940–1945, 2017.
32. G. Retsinas, N. Stamatopoulos, G. Louloudis, G. Sfikas and B. Gatos, "Nonlinear Manifold Embedding on Keyword Spotting Using t-SNE," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, 2017, pp. 487-492.
33. K. Pancerz, W. Paja and J. Gomuła, "Random forest feature selection for data coming from evaluation sheets of subjects with ASDs," 2016 Federated Conference on Computer Science and Information Systems (FedCSIS), Gdansk, 2016, pp. 299-302.
34. W. Cao, J. Xu and Z. Liu, "Speaker-independent speech emotion recognition based on random forest feature selection algorithm," 2017 36th Chinese Control Conference (CCC), Dalian, 2017, pp. 10995-10998
35. R. Gomes, M. Ahsan and A. Denton, "Random Forest Classifier in SDN Framework for User-Based Indoor Localization," 2018 IEEE International Conference on Electro/Information Technology (EIT), Rochester, MI, 2018, pp. 0537-0542.
36. Qing Song, Xiaoou Liu and Lu Yang, "The random forest classifier applied in droplet fingerprint recognition," 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Zhangjiajie, 2015, pp. 722-726.
37. Okfalisa, I. Gazalba, Mustakim and N. G. I. Reza, "Comparative analysis of k-nearest neighbor and modified k-nearest neighbor algorithm for data classification," 2017 2nd International conferences on

Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, 2017, pp. 294-298.

38. F. Guan, J. Shi, X. Ma, W. Cui and J. Wu, "A Method of False Alarm Recognition Based on k-Nearest Neighbor," 2017 International Conference on Dependable Systems and Their Applications (DSA), Beijing, 2017, pp. 8-12.
39. Y. Cai, H. Huang, H. Cai and Y. Qi, "A K-nearest neighbor locally search regression algorithm for short-term traffic flow forecasting," 2017 9th International Conference on Modelling, Identification and Control (ICMIC), Kunming, 2017, pp. 624-629.
40. X. Zhang, Bicheng Li and Xianzhu Sun, "A k-nearest neighbor text classification algorithm based on fuzzy integral," 2010 Sixth International Conference on Natural Computation, Yantai, 2010, pp. 2228-2231..
41. B. Waske and J. A. Benediktsson, "Fusion of Support Vector Machines for Classification of Multisensor Data," in IEEE Transactions on Geoscience and Remote Sensing, vol. 45, no. 12, pp. 3858-3866, Dec. 2007.
42. Liu Sheng, Wang Mengjun and Zhang Lanyong, "Research on information fusion of infrared and radar sensor based on SVM," Proceedings of 2012 International Conference on Measurement, Information and Control, Harbin, 2012, pp. 98-101.
43. T. Nakano, B. T. Nukala, S. Zupancic, A. Rodriguez, D. Y. C. Lie, J. Lopez, and T. Q. Nguyen, “Gaits Classification of Normal vs . Patients by Wireless Gait Sensor and Support Vector Machine (SVM) Classifier,” 2016 IEEE/ACIS 15th Int. Conf. Comput. Inf. Sci., pp. 1–6, 2016..
44. G. Jeong, P. H. Truong, and S. Choi, “Classification of Three Types of Walking Activities Regarding,” IEEE Sens. J., vol. 17, no. 9, pp. 2638–2639, 2017.
45. Y. Bengio, A. Courville, and P. Vincent, “Representation Learning : A Review and New Perspectives,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1798–1828, 2013.
46. U. M. M. Koldowski, “Spiking neural network vs multilayer perceptron : who is the winner in the racing car computer game,” Soft Comput., pp. 3465–3478, 2015.

47. Jędrzej Bieniasz, Mariusz Rawski, Krzysztof Skowron, Mateusz Trzepiński, "Evaluation of multilayer perceptron algorithms for an analysis of network flow data," Proc. SPIE 10031, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2016, 100314G (28 September 2016).
48. Lang B. (2005) Monotonic Multi-layer Perceptron Networks as Universal Approximators. In: Duch W., Kacprzyk J., Oja E., Zadrożny S. (eds) Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005. ICANN 2005. Lecture Notes in Computer Science, vol 3697. Springer, Berlin, Heidelberg.
49. A. Astorino and A. Fuduli, “The Proximal Trajectory Algorithm in SVM Cross Validation,” IEEE Trans. Neural Networks Learn. Syst., vol. 27, no. 5, pp. 966–977, 2016.
50. Q. Huang, J. Mao, and Y. Liu, “An Improved Grid Search Algorithm of SVR Parameters Optimization,” 2012 IEEE 14th Int. Conf. Commun. Technol., no. 2, pp. 1022–1026.
51. S. Chen and C. Liu, “Eye detection using discriminatory Haar features and a new efficient SVM,” Image Vis. Comput., vol. 33, pp. 68–77, 2015.
52. W. Shang, J. Cui, C. Song, J. Zhao, and P. Zeng, “Research on Industrial Control Anomaly Detection Based on FCM and SVM,” 2018 17th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. 12th IEEE Int. Conf. Big Data Sci. Eng., pp. 218–222, 2018.
53. Y. Xie and Y. Zhang, "An intelligent anomaly analysis for intrusion detection based on SVM," 2012 International Conference on Computer Science and Information Processing (CSIP), Xi'an, Shaanxi, 2012, pp. 739-742.
54. Y. Guang and N. I. E. Min, “Anomaly Intrusion Detection Based on Wavelet Kernel LS-SVM,” Proc. 2013 3rd Int. Conf. Comput. Sci. Netw. Technol., pp. 434–437, 2013.
55. C. Eswaran and R. Logeswaran, “A Comparison of ARIMA , Neural Network and Linear Regression Models for the Prediction of Infant Mortality Rate,” 2010 Fourth Asia Int. Conf. Math. Model. Comput. Simul., no. 2, pp. 34–39, 2010..

56. Friedman, Jerome H. et al. “Regularization Paths for Generalized Linear Models via Coordinate Descent.” Journal of statistical software 33 1 (2010): 1-22 ..
57. L. K. Jones, "Local Minimax Learning of Functions With Best Finite Sample Estimation Error Bounds: Applications to Ridge and Lasso Regression, Boosting, Tree Learning, Kernel Machines, and Inverse Problems," in IEEE Transactions on Information Theory, vol. 55, no. 12, pp. 5700-5727, Dec. 2009.
58. Kirpich A, Ainsworth EA, Wedow JM, Newman JRB, Michailidis G, McIntyre LM (2018) Variable selection in omics data: A practical evaluation of small sample sizes. PLoS ONE 13(6): e0197910..
59. [R. Bayindir, M. Ieee, M. Gok, and A. Dataset, “An Intelligent Power Factor Correction Approach Based on Linear Regression and Ridge Regression Methods,” 2011 10th Int. Conf. Mach. Learn. Appl. Work., vol. 2, pp. 313–315, 2011.
60. T. Verma, A. P. S. Tiwana, and C. C. Reddy, “Data Analysis to Generate Models Based on Neural Network and Regression for Solar Power Generation Forecasting,” 2016 7th Int. Conf. Intell. Syst. Model. Simul., pp. 97–100, 2016..

Appendix G: Open source code and data repositories for performance evaluation of eye gaze estimation systems

Manuscript Details

Manuscript number	SOFTX_2019_86
Title	Open source code and data repositories for performance evaluation of eye gaze estimation systems
Article type	Original software publication

Abstract

In contemporary eye gaze research, there is marked deficiency of open source software and datasets for quantitative and in depth evaluation of eye gaze data quality. To address this issue, an open source code repository named GazeVisual-Lib is developed and presented here, which contains a number of gaze data evaluation algorithms, visualizations and software tools that may be used to quantitatively analyse and evaluate the quality of gaze data from any consumer grade eye tracker. In addition, a new labelled eye gaze dataset collected from multiple user platforms and operating conditions is created, which can be used by eye gaze researchers and engineers to do benchmark comparison and evaluation of their gaze estimation systems.

Keywords	Eye gaze; data quality; performance evaluation; gaze dataset
Manuscript category	Domain Independent Tools and Technology
Corresponding Author	Anuradha Kar
Corresponding Author's Institution	National University of Ireland Galway
Order of Authors	Anuradha Kar, Peter Corcoran

Submission Files Included in this PDF

File Name [File Type]

Anuradha_Kar_SoftwareX_2019.docx [Manuscript File]

Anuradha_Kar_Conflict_of_Interest.pdf [Conflict of Interest]

Anuradha_Kar_Cover_letter.pdf [Supporting File]

Submission Files Not Included in this PDF

File Name [File Type]

GazeVisual_demo_video.mp4 [Video]

To view all the submission files, including those not included in the PDF, click on the manuscript title on your EVISE Homepage, then click 'Download zip file'.

Research Data Related to this Submission

Data set	https://data.mendeley.com/datasets/2txnw3y6gs/draft?a=b8ab5ca4-4d5e-463e-b5fc-a463d548102d

NUIG_EyeGaze01(A labelled benchmark eye gaze dataset)

NUIG_EyeGaze01(A labelled benchmark eye gaze dataset) is a rich and diverse gaze dataset, built using eye gaze data collected from a remote eye tracker under a wide range of operating conditions from three user platforms (desktop, laptop, tablet) from 20 participants. Gaze data is collected under one condition at a time.The dataset includes CSV files with gaze data and gaze angular variables, for 17 head poses, 4 user distances, 6 platform poses and 3 display screen size and resolutions.

Open source code and data repositories for performance evaluation of eye gaze estimation systems

Anuradha Kar *, Peter Corcoran
Department of Electrical & Electronic Engineering, National University of Ireland, Galway H91 TK33, Ireland; a.kar2@nuigalway.ie, peter.corcoran@nuigalway.ie

* Corresponding author: a.kar2@nuigalway.ie

Abstract.
In contemporary eye gaze research, there is marked deficiency of open source software and datasets for quantitative and in depth evaluation of eye gaze data quality. To address this issue, an open source code repository named GazeVisual-Lib is developed and presented here, which contains a number of gaze data evaluation algorithms, visualizations and software tools that may be used to quantitatively analyse and evaluate the quality of gaze data from any consumer grade eye tracker. In addition, a new labelled eye gaze dataset collected from multiple user platforms and operating conditions is created, which can be used by eye gaze researchers and engineers to do benchmark comparison and evaluation of their gaze estimation systems.

Keywords:
Eye gaze; data quality; performance evaluation; gaze dataset

1. *Motivation and Significance*

Eye gaze data quality refers to the validity of gaze data measured and reported by an eye tracker[1]. The common method for representing gaze data quality is by specifying gaze estimation accuracy which refers to the difference between the true and measured gaze positions [2]. There currently exists significant diversity in gaze accuracy measures as described in [3] and there are also no common or standard gaze accuracy metric, since different researchers define gaze accuracy independently. This leads to ambiguity in the interpretation of the quality of gaze data from different eye tracking systems and difficulty in comparison of two or more eye trackers. Moreover, with growing applications of eye gaze in unconstrained scenarios, e.g. in consumer electronics like augmented and virtual reality [4][5], handheld devices[6] and smart TVs[7], the eye trackers used in such use cases need to be thoroughly evaluated to ensure high quality and consistency of their gaze data outputs. This calls for the development and adoption of homogeneous metrics for reporting gaze accuracy and a consistent set of methods for complete characterization of eye trackers under different operating conditions[8].

There are several software tools [9]–[13]which have been developed by gaze researchers as well as eye tracker manufacturers for gaze data analysis, but all of them are concerned with determining eye movement characteristics (i.e. fixations, scanpath, saccades) and studying eye movement relationships with human cognitive process, such as creation of attention maps, understanding regions of user interests and visual search patterns. There are currently no software or open source tools which can be used by gaze researchers or engineers for quantitative evaluation of gaze data quality obtained from generic eye trackers.

Another aspect in gaze research is there currently exists no publicly available eye gaze dataset which contains gaze and ground truth data collected from users under different operating conditions of an eye tracker, from multiple user platforms. Without such datasets, it is difficult to interpret and compare the performance of any existing or new gaze estimation algorithm while they operate under unconstrained conditions, e.g. variable head poses, tracker orientations and user distances.

Keeping these aspects in mind, in this paper an open source code repository named GazeVisual-Lib (hosted on GitHub) is presented which contains a range of numerical and visualization methods for in-depth and quantitative analysis of eye gaze data quality. The methods included in the repository may be used on gaze data obtained from a generic/commercial remote eye tracker or eye tracking application. The methods are implemented as fully documented codes in Python language and can be used for a) de-noising and outlier removal of gaze data b) augmenting any gaze (fixation/scanpath) dataset by 7 different methods c) estimating gaze accuracy from input

gaze data in standard units of angular resolutions d) implementing several evaluation metrics and visualizations for exploration of gaze data quality[8]. Apart from these, the repository contains the source code of a desktop GUI software application (named GazeVisual, developed in Python language) which can implement the above and additional metrics and visualizations as well as interface with an eye tracker for live gaze data collection. The concept and components of this GUI software may be found in [14] where its utilities are discussed.

In addition to the methods described above, a new eye gaze dataset (named NUIG_EyeGaze01 (A labelled benchmark eye gaze dataset) hosted on Mendeley Data) is created using data from a commercial remote eye tracker while it operated on three different user platforms- a desktop, a laptop and a tablet under different operating conditions such as variable head poses, user distances, screen resolutions and platform poses. This data is made available and could be useful to gaze researchers and engineers for benchmark comparison of performance of other eye trackers, for building advanced gaze data evaluation metrics and also for understanding gaze error patterns caused by the different operating conditions. This dataset is also described in this work.

The motivation for developing the GazeVisual-Lib code repository is to present a set of standardized methods for gaze data evaluation to the interdisciplinary eye gaze research community, so that data characteristics from a variety of gaze tracking systems, applications and user platforms may be evaluated and compared under a unified framework of tools. Users can fully understand the sequence of development of these data evaluation methods as they are implemented in Python codes starting from raw gaze data, making these methods adaptable to gaze data from any source. Using these methods, the practical limits and capabilities of any gaze estimation system may be studied and compared quantitatively. Further, the desktop GUI application GazeVisual provided in the repository is intended for out-of-the-box use without requiring programming effort for evaluating gaze data. The purpose of releasing these methods in an open repository is to allow any gaze researcher or engineer to use and improve them as per their own requirements. With the rapidly progressing field of gaze research, such open source analysis tools are necessary that would allow adaptation of uniform gaze data evaluation methods as well as upgradation of the methods as per emerging requirements of new gaze based systems.

An experimental setting for using this code repository and its software components is schematically shown in Figure 1. For using the gaze data evaluation methods, first of all a sample of gaze data from one or more participants is required[8]. To collect gaze data: a) A user has to sit in front of the eye tracker under test, that is typically mounted on a computer screen and the user eyes are calibrated [15]. 2) The user is presented with visual stimuli on a graphical user interface (or UI as in Figure 1). The UI and eye tracker are started simultaneously and the eye tracker records the gaze coordinates of the user as the user gazes at the UI stimuli points on the screen. 3) The gaze data from the eye tracker is saved. The ground truth data which comprises of the screen coordinates of the UI stimuli points appearing during gaze data collection are also saved.

The raw gaze and ground truth data collected in the above manner can then be used as inputs to the codes in the repository and the GazeVisual software application. In the next section, the components of the *GazeVisual-Lib* repository are described. Details on how the gaze and ground truth data files may be used with the repository codes to estimate gaze accuracy, and successively implement other gaze data evaluation methods is provided.

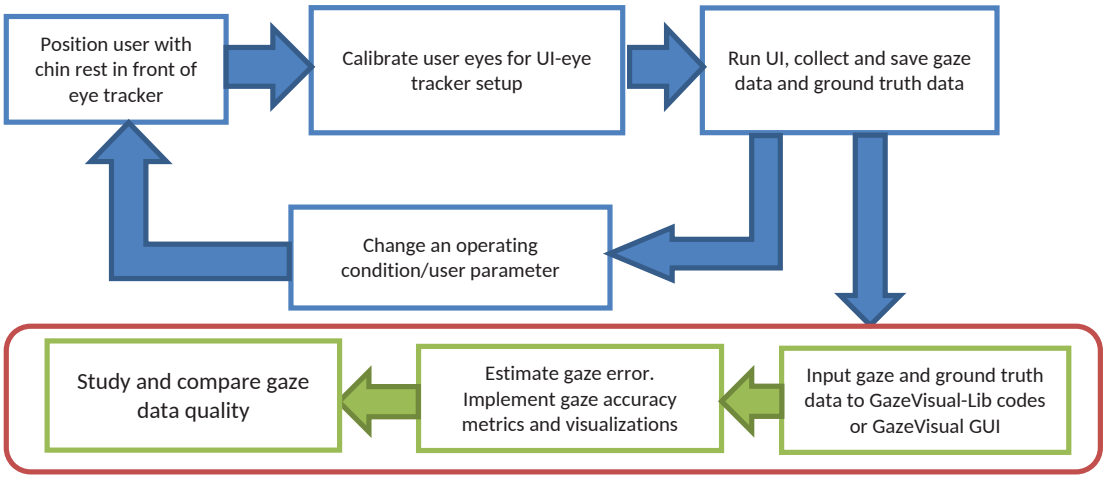


Figure 1: Workflow to implement the gaze data evaluation methods of the GazeVisual-Lib repository

2. *Description of GazeVisual-Lib and the new gaze data repository*

2.1 Organization of the repository

The GazeVisual-Lib repository has a set of folders containing numerical and visual methods, which require gaze data samples from an eye tracker and ground truth data (visual stimuli locations) as the main inputs along with values of setup variables like user-tracker distance, size and resolution of the display screen where gaze was tracked during data collection. The organization of the GazeVisual-Lib repository and its components is shown in Figure 2, followed by descriptions of their functionalities.

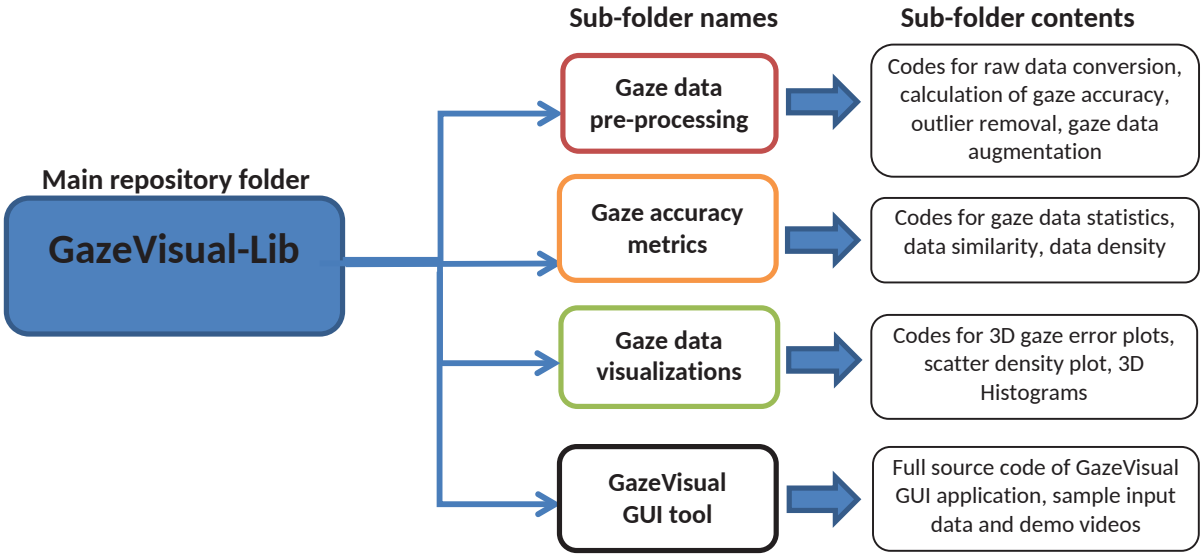


Figure 2: Organization of the GazeVisual-Lib code repository on GitHub

2.2 Functionalities of the GazeVisual-Lib repository components

The components of the GazeVisual-Lib repository are Python codes organized into four folders as shown in Figure 2. The contents and functionalities of the codes in each folder are described below.

A. "Gaze data pre-processing" folder

In this folder, there are three Python or .py files which are meant to perform the following functions: 1) Raw data conversion and calculation of accuracy: The main_proc.py file in this folder estimates gaze angular variables and accuracies from input raw gaze data and ground truth data (data samples are provided within the folder). The output of this Python code is a CSV file (named user_data_proc.csv, also present in the folder) which contains the estimated gaze angular variables (gaze yaw, pitch, frontal angle) and gaze accuracy values (angular differences between estimated gaze locations and stimuli locations). Full details of the calculations starting from raw gaze data till the values of gaze angular variables and accuracy are in our previous work [8]. 2) Outlier removal: Gaze data is almost always corrupted with outliers and it is impossible to observe any error patterns until outliers are removed. The outlier_removal.py file in this folder implements three different outlier detection and removal strategies which are 1D Median filtering, median absolute deviation and interquartile range method [8]. 3) Data augmentation: The code named data_augmentation.py in this folder implements six different methods for augmenting gaze data. These include addition of white and coloured noise, data interpolation, time-shifting, data convolution with cosine kernels and a combination of these. Augmented gaze datasets may be used for purposes such use of machine learning algorithms[16] to model gaze data patterns.

B. "Gaze accuracy metrics" folder

In this folder there is a sample gaze data file (user_data_proc.csv, which is the output file from the main_proc.py code described above) and three python files (data_statistics.py, data_similarity.py, scatter_density.py). These codes may be used to compute gaze data statistics, similarity between gaze datasets and gaze error spatial density using data from the user_data_proc.csv file. The data_statistics.py file calculates

mean, standard median absolute deviation, confidence intervals and Z-score[8] from an input gaze data column from the file. The data_similarity.py file computes similarity between gaze data, e.g. from different eye trackers. The similarity calculation is based on correlation, intersection and Bhattacharya distance [8] computed on histograms of two gaze datasets. The scatter_density.py file helps to create a gaze data density plot in which raw gaze data is plotted as data point clusters and color-mapped according to point densities, which helps to study gaze data patterns and detect any anomaly.

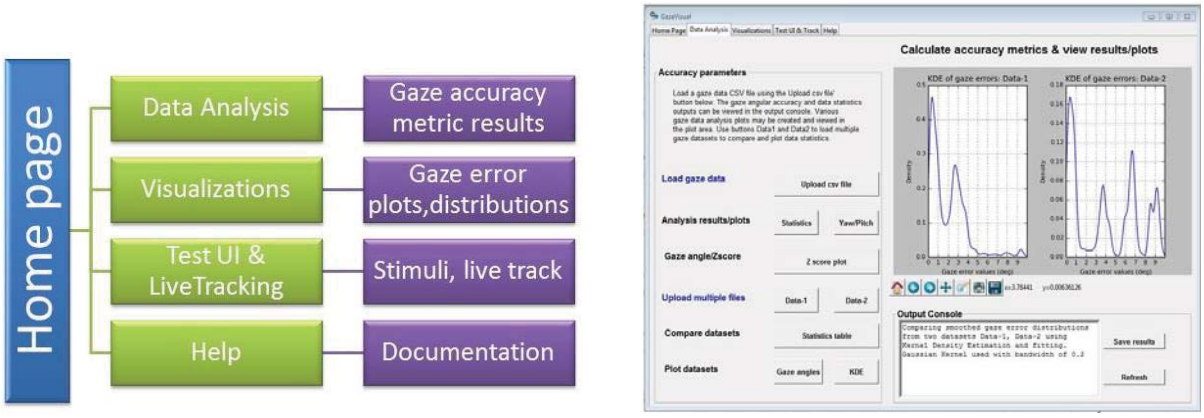
C. "Gaze data visualizations" folder

In this folder there is a sample gaze data file (user_data_proc.csv, output of the main_proc.py program) and three python codes that implement various visualizations as described in [8]. The file 3D_plot.py creates a 2D grid of on-screen stimuli locations and produces a 3D plot of the magnitude of gaze errors (plotted along Z axis) as a function of X and Y dimensions of the display screen. These plots help to diagnose gaze error levels over the display area. The eccentricity.py file plots creates a plot of gaze error levels mapped with respect to visual eccentricity or gaze angle (yaw, pitch) values on the display screen. This program may be used to study how gaze errors vary with visual angles, especially if user distance from the tracker is increased or decreased. [8]. The file 3D_histogram.py plots stacked 3D gaze error distributions using data from two or more trackers or experiments in a single plot. It helps to understand and compare data patterns, and gain insight into data characteristics such as where error values are concentrated or presence of data extremes.

D. "GazeVisual GUI tool" folder

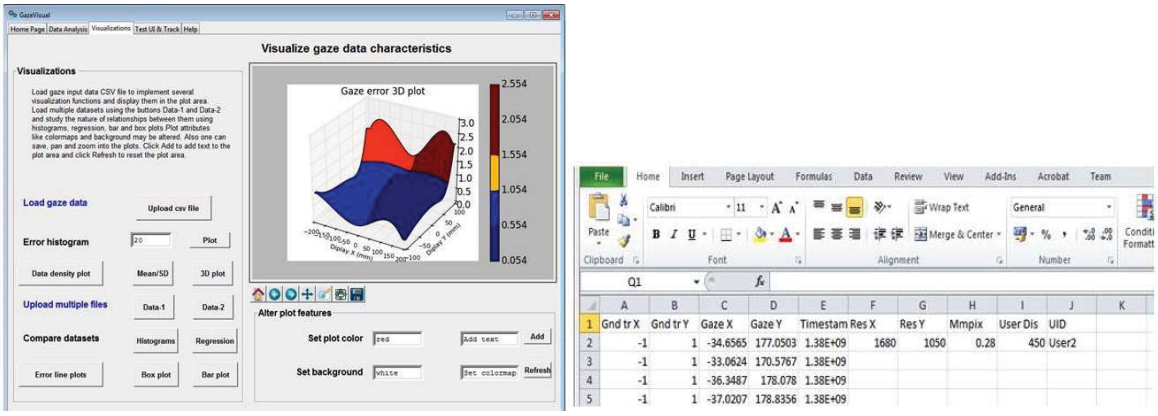
This folder contains the source code (gazevisual_v101.py) for the GazeVisual GUI application software. This software is designed for quick, easy and in-depth evaluation of eye tracker data, through a suite of statistical and visualization functions incorporated in it. GazeVisual comes in the form of a graphical user interface (GUI) and contains a range of functions to input and process gaze data files and produce various gaze accuracy metric results and visualizations. It can generate visual stimuli and can also be directly interfaced with an eye tracker to collect gaze data samples. It is entirely built in Python language using several data analysis and graphics libraries. The architecture of GazeVisual software is shown in Figure 3a and views of the software are in Figures 3b and 3c.

The GazeVisual software comprises of four independent windows containing a range of functions incorporated in it that are aimed towards gaze data evaluation. Input data format for the GazeVisual software is shown in Figure 3d below. It comprises of columns for raw gaze and ground truth data coordinates and input variables like display screen resolution and pixel pitch of the display (μ) [8] and user distance from the tracker. Two sample gaze data files that can be input to the GazeVisual software are provided in the GitHub folder. The GazeVisual software can be compiled as a generic Python program to produce the GUI application. Following this, the input gaze data files can be uploaded to the software using the "Upload csv file" button and then rest of the software functionalities may be implemented by using the other GUI buttons. Outputs of the software include gaze accuracy values, error statistics, plots of error numerical and spatial distributions and comparison of two gaze datasets. More details about the concept and functionalities of this software may be found in [14].



(a) (b)

Figure 3: (a) Architecture of the GazeVisual GUI software (b) View of the "Data Analysis" window of GazeVisual



(c) (d)

Figure 3: (c) View of the "Visualizations" window of the GazeVisual software (d)Input data format for GazeVisual.

The "Data Analysis" window provides functions for estimation of gaze angular variables, accuracy (in degrees), gaze error statistics and distributions using a single gaze data file, and allows comparison of these parameters for two gaze data files. The "Visualizations" window can be used to plot gaze error histograms and 3D spatial distributions from a gaze data file. It can also take in 2 gaze data files and compare their characteristics by creating correlation/regression and box plots. The Test UI& LiveTracking window can create static and dynamic stimuli for data collection from an eye tracker and also interface GazeVisual with an eye tracker for direct data collection.

The GazeVisual software has been tested with data from two remote eye trackers and a head-mounted eye tracker and has been seen to produce consistent results, provided input data formatting is done as in Figure 3d. Also, input gaze and ground truth data coordinates should have (0,0) values at the display screen center and must not contain NaN or non-numeric values. Sample input data files for testing this software may be found in the GitHub folder (e.g. "usr1_45_gazedata.csv"). GazeVisual can be compiled and run as a desktop application on any operating system having Python 2.7 with libraries such as Tkinter, Pygame, Statsmodels and Seaborn installed.

2.3 Description of the NUIG_EyeGaze01 gaze data repository

A rich and diverse gaze dataset, named NUIG_EyeGaze01 (A labelled benchmark eye gaze dataset) is built using gaze data collected under a wide range of operating conditions. This is a new kind of dataset, collected from three user platforms (desktop, laptop, tablet), from 20 participants under the influence of one condition at a time, with a commercial remote eye tracker. Informed consent from participants was obtained for all data collection sessions.

The operating conditions include, 17 head poses, 4 user distances, 6 platform poses and 3 display screen size and resolutions. Each gaze data file is labelled with the type of operating conditions under which it was collected. Each file belongs to one person, containing their gaze and ground truth data, gaze yaw, pitch angles and angular accuracy for one operating condition. The data collection procedure is presented in [8]. Organization of the dataset is shown in Figure 4. The full dataset may be obtained at present by contacting the authors and will also be released in the Mendeley open data repository in coming months. Detailed data descriptions are in the link to the repository at: https://data.mendeley.com/datasets/2txnw3y6gs/draft?a=6ca3629f-393c-4f80-93f6-8d56e6557792

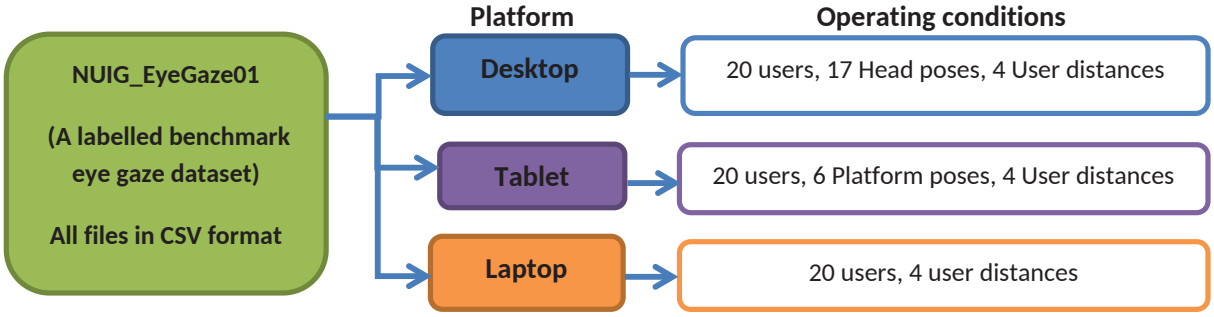


Figure 4: Dataset organization in the NUIG_EyeGaze01 repository on Mendeley data

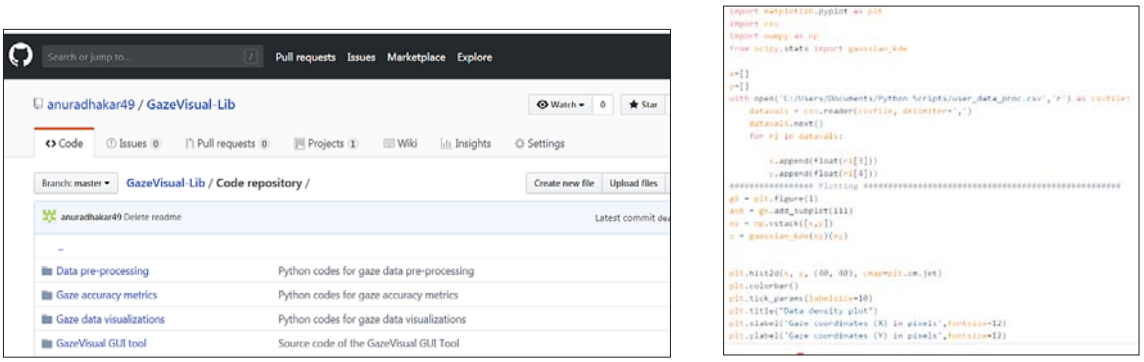
3. *Impact of the code and data repositories*

The open source gaze data evaluation methods of GazeVisual-Lib (released under GNU GPL 3.0 licence) could be useful for researchers, engineers and developers working with gaze estimation systems for thorough assessment of their gaze data quality. The methods could be especially beneficial for eye trackers which operate under unconstrained operating conditions where gaze data quality frequently becomes variable. Also, the GUI application GazeVisual may be used to perform prompt and in-depth gaze data evaluation without the need for any detailed programming knowledge. This could be particularly useful for the inter-disciplinary gaze research community where eye trackers are used widely in non-technological fields. The potential user group of GazeVisual-Lib is therefore quite diverse, ranging from gaze tracking system developers, researchers using eye trackers in virtual/augmented reality, human-computer interactions and cognitive sciences as well as generic users having any consumer grade eye tracker or gaze based application.

The new eye gaze database NUIG_EyeGaze01 presented in this paper could be beneficial to designers of gaze based systems for benchmark comparison of their system performances under challenging operating conditions such as variations of head pose, user distance and tracker orientations. These datasets could be also used for modelling and comparison of gaze error patterns, development and testing of gaze data anomaly detection algorithms or compensation algorithms for mitigating gaze errors.

4. *Illustrative example*

In order to implement an evaluation code from any of the folders of GazeVisual-Lib (Figure 5a), users need to copy the codes to their computers and have Python 2.7 and the imported libraries in the code installed. Then they can run the codes as normal python files. Sample data for running the codes is provided in the repository and information about each code is provided in their respective folders in the README files. A code snippet from the scatter_density.py file from the *"Gaze accuracy metrics"* folder is in Figure 5b. It loads the user_data_proc.csv file (which is provided within the folder) and creates a gaze data point cluster density plot. For running the GazeVisual GUI software, no installation is required and the source code can be run as described above. Sample data that can be used with the software are provided in the "GazeVisual GUI tool" folder. Links to several demo videos showing the functioning of the different windows of the software are provided in the "Sample videos" file of this folder.
https://github.com/anuradhakar49/GazeVisual-Lib/blob/master/Demo%20videos/Links%20to%20demo%20videos



(a) (b)

Figure 5: (a) View of the GazeVisual-Lib code repository (b) Snapshot of a python code from the repository

5. *Conclusions*

The code and data repositories presented in this work are intended to encourage adoption of standardized methods for analysing gaze data quality and practical performance evaluation of eye trackers under diverse operating conditions. The ultimate aim of these open resources is to enhance the usability and reliability of any gaze estimation system and their wide range of practical applications.

Acknowledgements

The research work presented here is funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded by FotoNation Ltd. Project ID: 13/SPP/I2868 on "*Next Generation Imaging for Smartphone and Embedded Platforms*".

References

[1] Holmqvist, K.; Nyström, M.; Mulvey, F. Eye tracker data quality: What it is and how to measure it. in Proc. ETRA'12, Santa Barbara, CA, USA, 28–30 March 2012; ACM: New York, NY, USA, 2012; pp. 45–52..

[2] Ooms, K.; Lapon, L.; Dupont, L.; Popelka, S. Accuracy and precision of fixation locations recorded with the low-cost Eye Tribe tracker in different experimental set-ups. J. Eye Mov. Res. 2015, 8, 1–24..

[3] Kar, A; Corcoran, P;. A Review and Analysis of Eye-Gaze Estimation Systems, Algorithms and Performance Evaluation Methods in Consumer Platforms. IEEE Access 2017, 5, 16495–16519.

[4] T. Piumsomboon, G. Lee, R. W. Lindeman and M. Billinghurst, "Exploring natural eye-gaze-based interaction for immersive virtual reality," 2017 IEEE Symposium on 3D User Interfaces (3DUI), Los Angeles, CA, 2017, pp. 36-39.

[5] J. Lee, H. Park, S. Lee, T. Kim and J. Choi, Design and Implementation of an Augmented Reality System Using Gaze Interaction, 2011 International Conference on Information Science and Applications, Jeju Island, 2011, pp. 1-8.

[6] Biedert, R, Dengel A, Buscher G, Vartan A, Reading and estimating gaze on smart phones. In Proc. ETRA '12, Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 385-388, 2012.

[7] Chen, Y., Chiang, C., Yu, C., Sun, W. and Yuan, S. "Real-time eye tracking and event identification techniques for smart TV applications," 2014 IEEE International Conference on Consumer Electronics - Taiwan, Taipei, 2014, pp. 63-64.

[8] Kar, A; Corcoran, P, Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations". Sensors 2018, 18, 3151.

[9] Špakov O., iComponent - Device-Independent Platform for Analyzing Eye Movement Data and Developing Eye-based Applications. Dissert. in Interactive Technol., vol. 9, University of Tampere (2008).

[10] Dalmaijer, E.S., Mathôt, S., & Van der Stigchel, S., PyGaze: an opensource, cross-platform toolbox for minimal-effort programming of eye tracking experiments. Behavior Research Methods, 46, 913-921, (2014)..

[11] Tula, A, et al. Heatmap Explorer: an interactive gaze data visualization tool for the evaluation of computer interfaces. In Proc. (IHC '16). ACM, New York, NY, USA, Article 24, pp1-9.

[12] van Renswoude, D.R., Raijmakers, M.E.J., Koornneef, A. et al. Behav Res (2018) 50: 834.

[13] Voßkühler, A.; Nordmeier, V.; Kuchinke, L. ,OGAMA (Open Gaze and Mouse Analyzer): Open-source software designed to analyze eye and mouse movements in slideshow study designs. Behav. Res. Methods 2008, 40, 1150..

[14] Kar, A; Corcoran, P, Gaze Visual - A Graphical Software Tool for Performance Evaluation of Eye Gaze Estimation Systems, in Proc 2018 IEEE (GEM), Galway, Ireland, 2018, pp. 1-9.

[15] Harezlak, K., Kasprowski, P., & Stasch, M. (2014). Towards accurate eye tracker calibration–methods and procedures. Procedia Computer Science, 35, 1073–1081.

[16] L. Li, Y. Wu, Y. Ou, Q. Li, Y. Zhou and D. Chen, Research on machine learning algorithms and feature extraction for time series. in Proc. IEEE PIMRC, Montreal, QC, 2017, pp. 1-5.

Table 1 – Code metadata

Nr	Code metadata description	*Please fill in this column*
C1	Current code version	*v1.0*
C2	Permanent link to code/repository used of this code version	https://github.com/anuradhakar49/GazeVisual-Lib
C3	Legal Code License	*GNU General Public License v3.0*
C4	Code versioning system used	*None*
C5	Software code languages, tools, and services used	*Python*
C6	Compilation requirements, operating environments & dependencies	*Operating environment : Python 2.7* *Dependencies: Python libraries* Tkinter, Pygame, Statsmodels, Seaborn , CSV, Pandas, Sklearn. Scipy, Numpy, Matplotlib, PIL
C7	If available Link to developer documentation/manual	*Concept of the gaze data evaluation methods may be found in the paper:* A. Kar, P. Corcoran: Performance Evaluation Strategies for Eye Gaze Estimation Systems with Quantitative Metrics and Visualizations. Sensors 18(9): 3151 (2018)
C8	Support email for questions	a.kar2@nuigalway.ie, peter.corcoran@nuigalway.ie

Appendix H: Eye Gaze for Consumer Electronics: Controlling and commanding intelligent systems

COURTESY OF FREEIMAGES.COM/IVA VILLI

Eye Gaze for Consumer Electronics

Controlling and commanding intelligent systems.

By Shabab Bazrafkan, Anuradha Kar, and Claudia Costache

Over the last several years, there has been much research and investigation on finding new ways to interact with the smart systems that currently form an integral part of our lives. One of the most widely researched fields in human–computer interaction (HCI) has been the use of human eye gaze as an input modality to control and command intelligent systems. For example, gaze-based schemes for hands-free input to computers for text entry/scrolling/pointing was proposed as early as in 1989 for disabled persons [1]. In the field of commercial applications, gaze-based interactions have brought immersive experiences in the world of virtual gaming and multimedia entertainment [2]. Eye gaze is also a significant feature for detecting the attention and intent of an individual. For example, gaze tracking could be implemented in a car to detect driver consciousness or in a smartphone to switch operations by sensing user attentiveness.

The concept of eye-gaze estimation in unconstrained conditions has become a hot research topic in the last few years, with a variety of algorithms and setup configurations being developed through interdisciplinary research. Much of the initial research involved intrusive systems for gaze tracking, such as electrooculography, scleral coils, or special contact lenses, requiring the user to wear some kind of head-mounted device/eye attachment to estimate his or her gaze. Such systems are awkward for the user and generally unsuited for use in consumer

Digital Object Identifier 10.1109/MCE.2015.2464852
Date of publication: 29 October 2015

One of the most widely researched fields in human–computer interaction has been the use of human eye gaze as an input modality to control and command intelligent systems.

devices and systems. No one wants to start looking like a Borg to control the TV set!

More recent research has employed passive video-based gaze trackers placed at a specific distance from the user, and gaze estimation is effected by capturing and processing images of the full face or eye region in natural light or using active illumination at near infrared (NIR) wavelengths. The NIR illumination is not perceived by the subject, and such systems are nonintrusive. Apart from improved user convenience, another advantage of these passive systems is their low setup cost, as they just use one or two cameras and a few LEDs [3].

Despite significant efforts by researchers in this field, the usability of gaze cues in the consumer domain remains marginal in terms of accuracy and reliability, and there are additional issues surrounding the suitability of eye gaze for calibration-free use cases.

In this article, we discuss some existing applications that use human eye gaze as a vital cue for various consumer platforms. For each of the use cases, the utility and advantages as well as inherent limitations are discussed.

METHODS FOR DETERMINING THE EYE GAZE POINTS

In the following sections, we cover some of the most interesting approaches developed for video-based gaze trackers and their implementation. Eye-gaze estimation methods can be broadly categorized into two general approaches: 1) appearance-model-based and 2) feature-based methods. We give a brief description for each of these approaches in the next sections.

APPEARANCE-MODEL-BASED METHODS

The appearance-model-based methods use the general shape of the eyes and position of the pupils relative to the eye corners to find the point of gaze [4]–[6]. A pretrained model of the shape and appearance (texture and lighting) of the eye region is fitted to a sequence of image frames [2], wherein it provides a fit to eye regions depending on whether the model has been trained with sufficient data that match the acquisition conditions and physical characteristics of the input eye region(s).

ADVANTAGES AND DISADVANTAGES OF THE APPEARANCE-MODEL-BASED METHODS

The main advantage of these approaches is their low hardware requirements. Usually, methods in this category are suitable for implementation on platforms without a high-resolution camera or additional light sources.

The main disadvantage of this class of methods is low accuracy, typically being 2 or 3° for fixed head position [4], [5], [7]. As a result, it can be stated that small changes in eye gaze (1–2° difference), which can be important for smartphone applications, may not be detected by the appearance-based methods (depending on size of the smartphone, the angle spanned by the pupil can be very small—7–10°). Other problems arise with changes in head pose, variation in global illumination due to different directional light sources, facial expressions, and skin color. Compensating for all of these issues requires large training data sets and makes appearance-based methods computationally expensive.

FEATURE-BASED METHODS

These methods take into account various characteristics of the human eye to identify a set of distinctive features like contours (limbus and pupil contour), eye corners, and cornea reflections of NIR illuminators (LEDs). These approaches

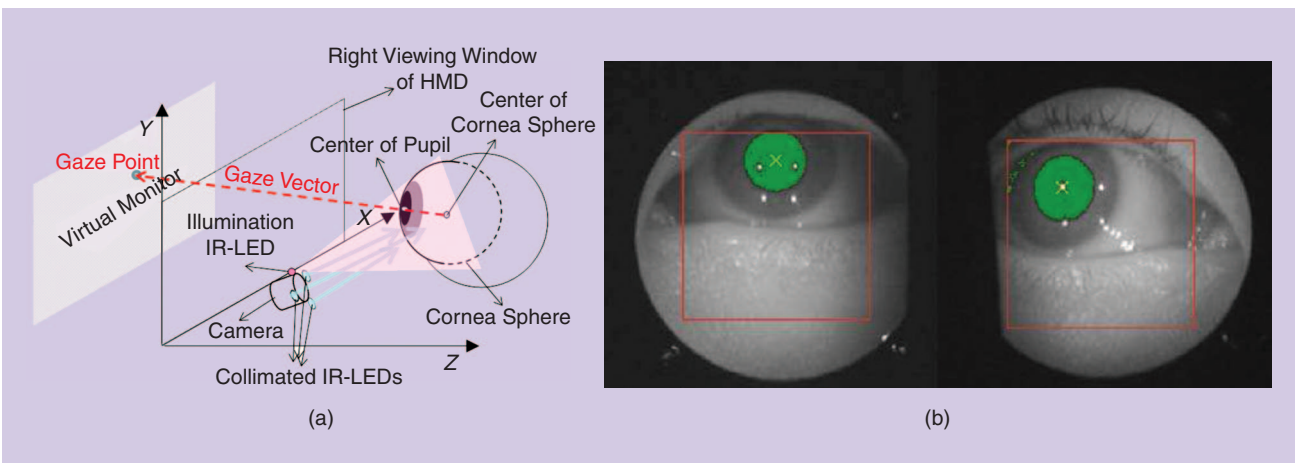


FIGURE 1. (a) A schematic for a feature-based eye-gaze-tracking method [27]. (b) Tracking NIR reflections on the cornea and pupil center in the PCCR method [17].

Table 1. A summary of hardware setup requirements and accuracies achieved by various gaze-tracking methods in the literature.

Method	Required Setup	Accuracy
Appearance model [13]	Webcam (720 × 576) chinrest, distance: 75 cm	2–5°
Appearance model [2]	VGA (640 × 480) distance: 40–70 cm	4°
Appearance model [4]	Webcam (1,280 × 1024) 50–60 cm	1° fixed head 2° slight motion
Feature (eye-model) based [14]	One camera (8 MP), 25–30 LEDs	1°
Feature (model) based [15]	Two cameras, two rings of LEDs for each camera	1°
Feature (regression) based [16]	Camera (1,024 × 768) one NIR source, 71 cm	0.4° head fixed, 1° slight head pose

can further be subdivided into two main branches: regression-based and model-based methods [8].

REGRESSION-BASED METHODS

In these methods (also called pupil center corneal reflection [PCCR] techniques), the vector between the corneal reflections and pupil center is tracked and mapped geometrically with a polynomial regression function to gaze coordinates on screen [Figure 1(a)].

MODEL-BASED METHODS

The eye-model-based techniques use the geometrical model of the human eye [Figure 1(b)] along with NIR light sources, multiple cameras, and one or more mirrors to develop accurate ray tracing to estimate gaze direction [9]–[11].

PROS AND CONS OF FEATURE-BASED METHODS

The advantages of feature-based methods include their higher accuracy in comparison to the appearance-based methods, as shown in Table 1. The main disadvantage is the hardware requirements for implementing these techniques as they need several light sources or multiple cameras. The other problem is that if the NIR glints in the cornea disappear due to head movements or additional glints appear in the eye from other NIR sources (e.g., halogen lights, sunlight, and reflections from eyeglasses), the method will fail.

> The prime hindrances to such gaze-based password-entry systems are the possibilities of identity theft and that unintentional eye and body movements may lead to erroneous gaze points and incorrect password entry.

CALIBRATION IN EYE-GAZE ESTIMATION

Since the eye-gaze systems should detect the human's optical axis but the actual point of gaze is defined by human visual axis and the difference between these two axes is specific for each person (it can be as high as 5° horizontally and 1.5° vertically [12]), the gaze-tracking system should be calibrated for each individual as an integral part of the tracking procedure. Calibration is done by looking at certain predetermined points and is often a cumbersome process. The error in the calibration procedure is added to other sources of error, which usually leads to significant overall tracking error.

APPLICATIONS OF EYE GAZE IN CONSUMER ELECTRONIC DEVICES

EYE-GAZE TRACKING IN INTELLIGENT DRIVING ASSISTANCE SYSTEMS

Reports from agencies like the U.S. National Highway Traffic Safety Administration stating that the majority of fatal auto accidents are due to driver inattention have sparked interest in

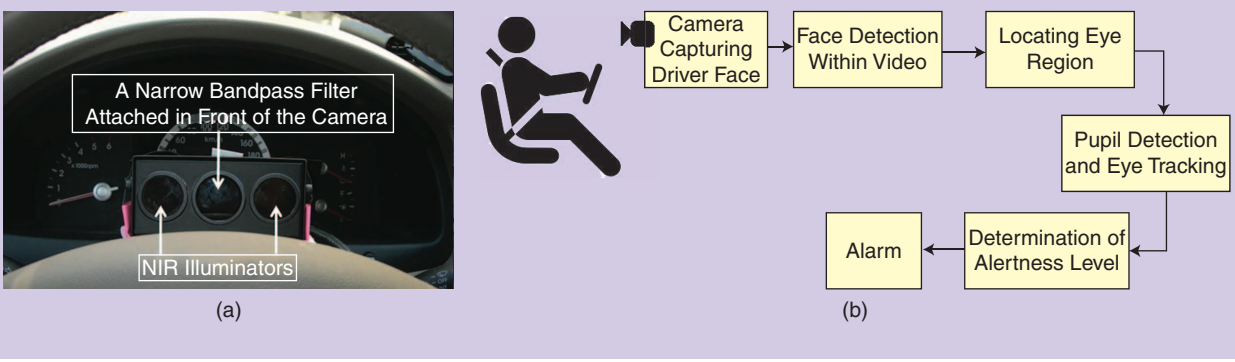


FIGURE 2. (a) A gaze-tracking setup in a car [26]. (b) A schematic of an appearance-based method for gaze tracking in a driver alert system.

The key issues in the usability of such gaze-based driver-assistance systems include inherent problems in eye detection due to variable illumination, occlusion of the eye region due to shadows or eyeglasses, false alarms, and real-time operability.

industry and academia in developing advanced driver monitoring and assistance systems [26]. Leading car manufactures like Toyota and Nissan have long been involved in developing interactive driver-support systems based on actively tracking driver gaze or blinking patterns to evaluate driver vigilance and drowsiness levels [2]. The setups for cars use machine-vision algorithms (e.g., percentage of eyelid closure time) active appearance models, or conventional PCCR techniques [17], comprising cameras and infrared light sources mounted on the car's dashboard [Figure 2(a)].

MAJOR LIMITATIONS

The key issues in the usability of such gaze-based driver-assistance systems include inherent problems in eye detection due to variable illumination, occlusion of the eye region due to shadows or eyeglasses, false alarms, and real-time operability.

SMARTPHONE/HANDHELD PLATFORMS USING GAZE AS AN INPUT MECHANISM

Despite smartphone releases from Samsung and Apple rumored to have functions for eye gaze, these inventions remain within the confines of patents; they have not yet been commercially realized. The proposed utilities of using eye gaze as an input mechanism as mentioned in the patents (e.g., Samsung's *Method and Apparatus for User Interface Using Gaze Interaction,* US2014/0232638A1 and Apple's *Electronic Devices With Gaze Detection Capabilities,* US2013/0135198A1) include user gaze commands for unlocking the cell phone, dimming the display screen, pausing/resuming video playback, suspending various sensors, or reducing the number of processing cores utilized by circuitry to optimize device power consumption.

MAJOR LIMITATIONS

Handheld devices like smartphones and tablets are usually dynamic platforms where the conditions for stable eye tracking are highly challenged due to variable relative positions of the user with respect to the device, unpredictable hand jitter and movements, unstable illumination, and the "Midas-touch" effect [2], [17].

DOMOTIC CONTROLS AND IPTV

Eye-gaze-based domotic controls are a new class of human–environment interaction methodologies [18]. These have been inspired by the increasing requirements of self-sufficiency and autonomy for designing smart homes that ensure improved quality of living—especially for the elderly and people with disabilities [20]. Another recently commercialized gaze-oriented paradigm includes gaze-controlled IPTV—brought recently to market by Haier in conjunction with Tobii—which uses the PCCR technique for gaze localization [Figure 3(a)]. Furthermore, a patent by Park et al. (*Gaze Tracking System and Method for Controlling Internet Protocol TV at a Distance,* US20120133754A1) enumerates the possible utilities of such an IPTV, e.g., moderating hue, brightness, and TV display depending on user gaze and understanding user interest [19].

MAJOR LIMITATIONS

The typical limitations of such gaze-based assisted-living systems comprise the inability of tracking systems to distinguish

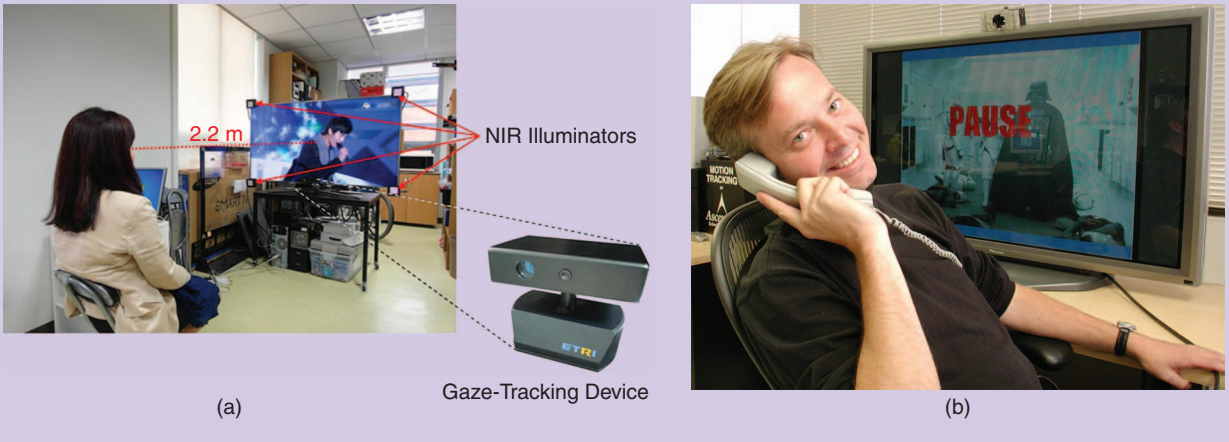


FIGURE 3. (a) The Haier IPTV setup with Tobii user tracker in front. (Photo used with permission from [31].) (b) An attentive TV that senses engagement [20].

FIGURE 4. (a) An eye-gaze plot showing how user attention is attracted by banners on a website [29]. (b) A gaze heat map showing that videos on a webpage attract maximum attention. (Photos courtesy of simpleusability.com.)

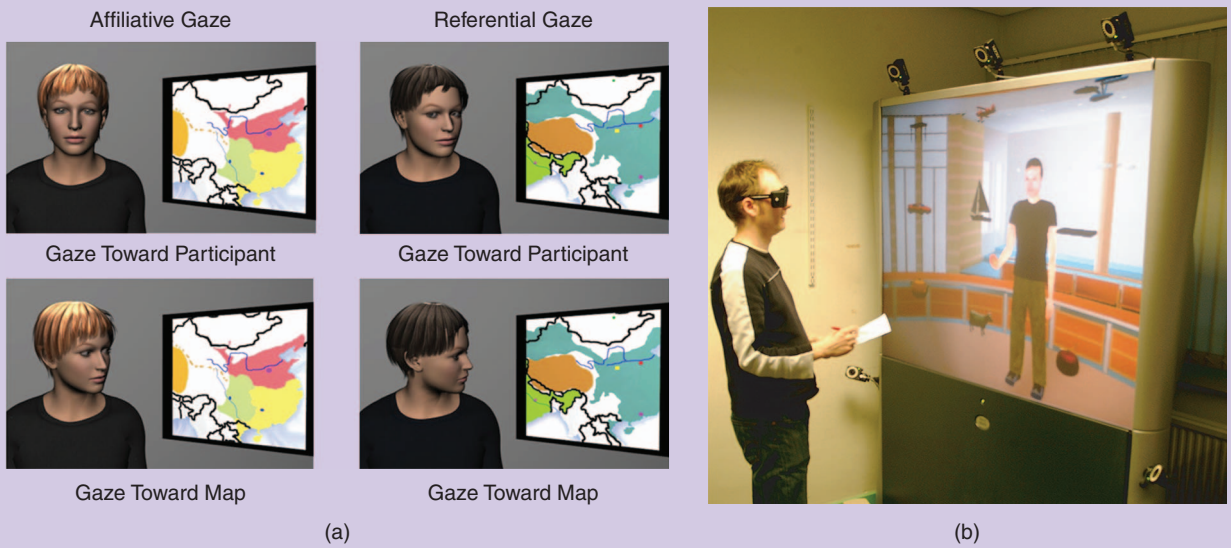


FIGURE 5. (a) A virtual lecture delivered with various gaze patterns [28]. (b) Eye-gaze tracking in a virtual avatar [30].

gaze from passing glances and the requirement for specialized hardware and simple user interfaces due to difficulty in handling complex tasks by gaze gestures.

GAZE TRACKING IN MARKETING AND E-COMMERCE

Innovative ways of studying consumer attention and response toward online products and advertisements are emerging through tracking user eye-gaze patterns and mapping them into gaze plots and heat maps that reveal salient information about a brand or its representation [24]. Interesting results from such user-gaze surveys include statistical methods for reorganization of web page content to maximize user attention as well as information about the relative significance of images and videos in a web page, restricting the quantity of advertisements per page, and enabling suitable social media content (e.g., Tweets and "likes") to increase consumer focus (Figure 4).

MAJOR LIMITATIONS

Fundamental issues include the fact that gaze maps may not reveal anything about the higher-level processes of user

attention or intent, invalidating the results. Long fixations could indicate either attraction to an element or difficulty in understanding it. It also requires specialized and expensive setup.

GAZE TRACKING IN VIRTUAL AND AUGMENTED REALITY

Virtual characters and environments have crossed the realm of gaming [2] to find use in interactive applications such as virtual tourism, urban planning, and online education (Figure 5). To enhance user experience in such advanced virtual reality (VR) applications, user gaze is actively tracked in real time to bring scenes into sharp focus by determining the user point of view in three-dimensional space, facilitate user navigation by deciphering visual attention, and controlling a virtual gamer's responses according to speed of eye movements. Important contributors to this field are Sony, Nintendo, and Sensor Motoric Instruments (SMI) GmBH, whose gaze-coupled VR applications extend from gaming to gaze interactions and ergonomics [21], [22].

Overall, it can be stated that despite decades of research in this field, the realization of a winning implementation that provides both reliable and accurate gaze-controlled interfaces in the consumer domain is still some distance in the future.

MAJOR LIMITATIONS

Key issues in eye tracking for VR and augmented reality involve Midas touch, gaze-point tracker inaccuracies leading to incorrect actions/selection, and uncertainties in tracking resulting from miscalibration or latency.

EYE GAZE FOR SECURITY AND AUTHENTICATION

Enhanced resistance against shoulder-surfing attacks has encouraged the development of gaze-based password schemes [25]. While the simplest of such authentication methods include entering alphanumeric characters as a security PIN with eye gaze, advanced security options involve gaze-image passwords, i.e., password entry by gazing with a specific sequence on an image on a screen or at multiple features in an image [23].

MAJOR LIMITATIONS

The prime hindrances to such systems are the possibility of identity theft and that unintentional eye and body movements may lead to erroneous gaze points and incorrect password entry.

CONCLUDING THOUGHTS

The potential of gaze-based interactions in modern consumer devices and platforms is still being explored. Major challenges exist due, in particular, to the highly dynamic modalities and unconstrained use cases for handheld devices and in the automotive sector due to the rigorous safety considerations and the wide range of lighting conditions experienced in the vehicle cockpit. The requirements for a consumer-grade gaze-tracking system include real-time, high-accuracy gaze estimation, easy calibration (calibration-free is preferable), and robust responses to widely varied lighting conditions. Robustness to head pose is also important, although this is a more tractable problem if a high-accuracy head model is available. As in all consumer electronics system designs, it is essential to maximize system performance while reducing unit costs.

From this point of view, two of the most powerful methods for gaze estimation developed to date are presented in this review, appearance-based and feature-based techniques. While each method has its respective advantages and drawbacks, the appearance-based models, because of their lower hardware requirements, would appear to offer a more feasible solution for mobile platforms. However, they do suffer from lower accuracy and are sensitive to changes in head pose, illumination, and facial expression. Alternatively, feature-based or corneal-glint methods can provide significantly higher accuracies, but due to a requirement for complex hardware, stable positioning, and detailed calibration, they are really only suited for indoor and desktop environments.

Overall, it can be stated that despite decades of research in this field, the realization of a winning implementation that provides both reliable and accurate gaze-controlled interfaces in the consumer domain is still some distance in the future. In fact, much of the research in this field only serves to emphasize that the accurate and nonintrusive measurement of eye gaze in real time and in unconstrained use cases using consumer-grade technology remains a major challenge.

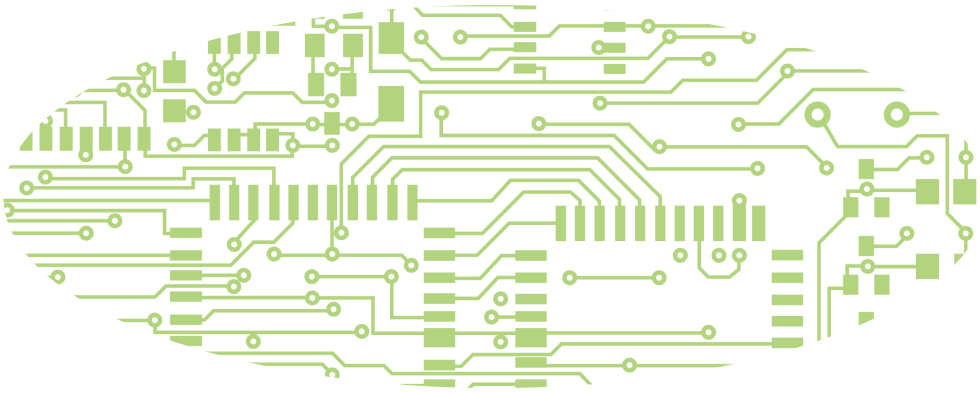
More importantly, the ultimate prospect to employ eye gaze as a means of HCI depends entirely on a very nontechnical factor, and one that is extremely difficult to quantify—the simplicity and ease-of-use of this sophisticated technology. Unless it can be realized in a robust, reliable, and calibration-free form, then consumers are unlikely to engage with and adopt the technology. And without the magic of simplicity, it is unlikely to become a sustainable success and a new and practical mode of HCI.

ABOUT THE AUTHORS

Shabab Bazrafkan (sbazrafkan@fotonation.com) is with the Center for Cognitive, Connected, and Computational Imaging, National University of Ireland, Galway.

Anuradha Kar (akar@fotonation.com) is with the Center for Cognitive, Connected, and Computational Imaging, National University of Ireland, Galway.

Claudia Costache (ccostache@fotonation.com) is with the Center for Cognitive, Connected, and Computational Imaging, National University of Ireland, Galway.



REFERENCES

[1] T. E. Hutchinson, K. P. White, W. N. Martin, K. C. Reichert, and L. A. Frey, "Human–computer interaction using eye-gaze input," *IEEE Trans. Syst. Man Cyber.*, vol. 19, no. 6, pp. 1527–1534, 1989.

[2] P. M. Corcoran, F. Nanu, S. Petrescu, and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," *IEEE Trans Consum. Electron.*, vol. 58, no. 2, pp. 347–355, 2012.

[3] A. K. Bhowmik, *Interactive Displays: Natural Human-Interface Technologies*. Hoboken, NJ: Wiley, 2014.

[4] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, "Adaptive linear regression for appearance-based gaze estimation," *IEEE Trans. Pattern Anal. Mach. Intell,* vol. 36, no. 10, pp. 2033–2046, 2014.

[5] Y. Sugano, Y. Matsushita, and Y. Sato, "Appearance-based gaze estimation using visual saliency," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 329–341, 2013.

[6] F. Lu, T. Okabe, Y. Sugano, and Y. Sato, "Learning gaze biases with head motion for head pose-free gaze estimation," *Image Vis. Comput.*, vol. 32, no. 3, pp. 169–179, 2014.

[7] Y.-T. Lin, R.-Y. Lin, Y.-C. Lin, and G. C. Lee, "Real-time eye-gaze estimation using a low-resolution webcam," *Multimed. Tools Appl.*, vol. 65, no. 3, pp. 543–568, 2013.

[8] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, 2010.

[9] C.-C. Lai, S.-W. Shih, and Y.-P. Hung, "Hybrid method for 3-D gaze tracking using glint and contour features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 1, pp. 24–37, 2015.

[10] J. Chen and Q. Ji, "A probabilistic approach to online eye gaze tracking without explicit personal calibration," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 1076–1086, 2015.

[11] A. Al-Rahayfeh and M. Faezipour, "Application of head flexion detection for enhancing eye gaze direction classification," in *Proc. 36th Annu. IEEE Int. Conf. Engineering Medicine Biology Society (EMBC),* 2014, pp. 966–969.

[12] E. D. Guestrin, and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1124–1133, 2006.

[13] R. Valenti, N. Sebe, and T. Gevers, "Combining head pose and eye location information for gaze estimation," *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 802–815, 2012.

[14] H. Lee, N. Iqbal, W. Chang, and S.-Y. Lee, "A calibration method for eye-gaze estimation systems based on 3D geometrical optics," *IEEE Sensors J.*, vol. 13, no. 9, pp. 3219–3225, 2013.

[15] Y. Ebisawa and K. Fukumoto, "Head-free, remote eye-gaze detection system based on pupil-corneal reflection method with easy calibration using two stereo-calibrated video cameras," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 10, pp. 2952–2960, 2013.

[16] L. Sesma-Sanchez, A. Villanueva, and R. Cabeza, "Gaze estimation interpolation methods based on binocular data," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 8, pp. 2235–2243, 2012.

[17] R. I. Hammoud, Ed. (2015, July 8). *Passive Eye Monitoring—Algorithms, Applications and Experiments.* [Online]. Available: http://www.springer.com/us/book/9783540754114

[18] D. Bonino, E. Castellina, F. Corno, and L. De Russis, "DOGeye: Controlling your home with eye interaction," *Interact. Comput.*, vol. 23, no. 5, pp. 484–498, 2011.

[19] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, "Gaze tracking system at a distance for controlling IPTV," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[20] J. S. Shell, R. Vertegaal, and A. W. Skaburskis, "EyePliances," in *CHI '03 Extended Abstracts on Human Factors in Computing Systems,* 2003, p. 770.

[21] D. A. Bowman and L. F. Hodges, "Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments," *J. Vis. Lang. Comput.*, vol. 10, no. 1, pp. 37–53, 1999.

[22] T. Starner, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R. W. Picard, and A. Pentland, "Augmented reality through wearable computing," *Presence Teleoperators Virtual Environ.*, vol. 6, no. 4, pp. 386–398, 1997.

[23] D. Rozado, "Using gaze based passwords as an authentication mechanism for password input," in *Proc. 3rd Int. Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction,* Lund, Sweden, 2013.

[24] J. W. Owens and B. S. Chaparro, "Text advertising blindness: The new banner blindness?" *J. Usability Stud.*, vol. 6, pp. 172–197, 2011.

[25] J. Weaver, K. Mock, and B. Hoanca, "Gaze-based password authentication through automatic clustering of gaze points," in *Proc. IEEE Int. Conf. Systems, Man, Cybernetics*, Oct. 2011, pp. 2749–2754.

[26] J. Jo, S. Lee, H. Jung, K. Park, and J. Kim, "Vision-based method for detecting driver drowsiness and distraction in driver monitoring system," *Opt. Eng.* 0001, vol. 50, no. 12, pp. 127202–127224.

[27] E. Lee and K. Ryoung Park, "A study on eye gaze estimation method based on cornea model of human eye," *Lecture Notes in Computer Science*, vol. 4418, 2007, pp. 307–317.

[28] K. Ruhland, C. E. Peters, S. Andrist, J. B. Badler, N. I. Badler, M. Gleicher, B. Mutlu, and R. McDonnell, "A review of eye gaze in virtual agents, social robotics and HCI: behaviour generation, user interaction and perception," *Computer Graphics Forum*, 2015.

[29] A. J. Ramakrisnan, F. H. A. Razak, and D. A. Ramba, "Evaluation of user Interface Design for Leaning Management System (LMS): Investigating student's eye tracking pattern and experiences," *Procedia – Soc. Behav. Sci.*, vol. 67, pp. 527–537, 2012.

[30] D. R. Murray,, P. Sharkey, A. Steed, P. Dickerson, and J. Rae, "An assessment of eye-gaze potential within immersive virtual environments," *ACM Trans. Multimedia Comput., Commun. Appl.*, vol. 3, no. 4, pp. 1–17, 2007.

[31] H. C. Lee, W. O. Lee, C. W. Cho, S. Y. Gwon, K. R. Park, H. Lee, and J. Cha, "Remote gaze tracking system on a large display," *Sensors*, vol. 13, no. 10, 2013, pp. 13439–13463.

CE

Appendix I: Convolutional Neural Network Implementation for Eye-Gaze Estimation on Low-Quality Consumer Imaging Systems

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2899869, IEEE Transactions on Consumer Electronics

Convolutional Neural Network Implementation for Eye-Gaze Estimation on Low-Quality Consumer Imaging Systems

Joseph Lemley, *Student Member, IEEE,* Anuradha Kar, *Student Member, IEEE,* Alexandru Drimbarean, *Member, IEEE,* and Peter Corcoran, *Fellow, IEEE*

***Abstract*—Accurate and efficient eye gaze estimation is important for emerging consumer electronic systems such as driver monitoring systems and novel user interfaces. Such systems are required to operate reliably in difficult, unconstrained environments with low power consumption and at minimal cost. In this paper a new hardware friendly, convolutional neural network (CNN) model with minimal computational requirements is introduced and assessed for efficient appearance-based gaze estimation. The model is tested and compared against existing appearance-based CNN approaches, achieving better eye gaze accuracy with significantly fewer computational requirements.**

***Index Terms*—Eye gaze, Neural Networks, Deep Learning**

I. Introduction

The potential of eye gaze tracking and gaze-based human computer interactions in modern consumer devices is currently an active topic for exploration. Eye gaze has been used to derive human behavioral cues, as an input modality and for achieving immersive user experiences in virtual and augmented reality systems. However, applications of gaze in consumer devices operating in real world conditions face tough challenges in terms of accuracy and reliability.

A. Gaze Tracking in Consumer Devices

After decades of research on desktop-based gaze estimation techniques, the focus has recently shifted to building eye gaze applications for dynamic platforms such as driver monitoring systems [1] and handheld devices [2]. For an automobile driver, eye-based cues such as levels of gaze variation, speed of eyelid movements and eye closure can be indicative of a driver's cognitive state. These can be useful inputs for intelligent vehicles to understand driver attentiveness levels, lane change intent, and vehicle control in the presence of obstacles to avoid accidents [3]. Handheld devices like smartphones and tablets form unique platforms for gaze tracking applications wherein gaze may be used as an input modality for device control, activating safety features and novel user interface (UI) designs [4].

The most challenging aspect of these modern gaze applications includes operation under dynamic user conditions and unconstrained environments. Further requirements for implementing a consumer-grade gaze tracking system include real-time high-accuracy operation, minimal or no calibration, and robustness to user head movements and varied lighting conditions. Therefore accurate and reliable gaze tracking typically demands high quality cameras and special equipment like narrow angle lenses, external illumination, and stereo setups for capturing eye region features with sufficient details. As a result, gaze estimation systems frequently become costly with complicated setups, which are unsuitable for generic and consumer applications.

Therefore a major challenge of gaze based consumer electronics design involves maximizing system performance while reducing costs and system complexities.

B. Deep Learning for Eye Gaze

In this paper, we introduce a calibration-free method for appearance-based gaze estimation that is suitable for consumer applications and low cost hardware with real time requirements, using a Convolutional Neural Network (CNN). CNNs were popularized by Lecun et al. [5], who used them successfully for handwritten digit classification. These networks are inspired by the organization of the visual cortex and allow spatial information to be more efficiently learned. Convolutional Neural Networks can be used on input with any number of dimensions, but due to their success in pictures, are most popularly implemented for 2D input plus color channels. Other popular types of CNNs include 1D CNNs, which are commonly used for time series, and 3D CNNs, which can be used for volumetric data or time series data where the third dimension represents either spatial frames or temporal frames [6]. Although CNNs have become ubiquitous for most computer vision tasks, they have yet to become popular for eye gaze estimation.

Neural network implementations are particularly important for embedded and low-power consumer imaging systems because hardware based CNNs can provide significant improvement in power efficiencies over CPU/GPU based solutions.

Manuscript received June 8, 2018. Revised September 09, 2018. 2nd revision December 15, 2018. 3nd revision January 27, 2019

This research is funded under the SFI Strategic Partnership Program by Science Foundation Ireland (SFI) and FotoNation Ltd. Project ID: 13/SPP/I2868 on Next Generation Imaging for Smartphone and Embedded Platforms and by an Irish Research Council Employment Based Programme Award. Project ID: EBPPG/2016/280.

J. Lemley is with the Department of Electrical and Electronic Engineering, NUI Galway, Galway Ireland (e-mail: J.lemley2@nuigalway.ie)

A. Kar is with the Department of Electrical and Electronic Engineering, NUI Galway, Galway Ireland (e-mail: a.kar2@nuigalway.ie)

A. Drimbarean is with Fotonation ltd, Galway Ireland (e-mail: Alexandru.Drimbarean@xperi.com)

P. Corcoran is with the Department of Electrical and Electronic Engineering, NUI Galway, Galway Ireland (e-mail: peter.corcoran@nuigalway.ie)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2899869, IEEE Transactions on Consumer Electronics

A useful comparison of the performance gains of Tensor Processing Units (TPU), a custom Application Specific Integrated Circuit (ASIC) running deep neural networks, with conventional GPU arrays is given in [7]. The TPU is on average about 15X - 30X faster than its contemporary GPU or CPU, with TOPS/Watt about 30X - 80X higher depending on the percent usage of the TPU. While the use case described in [7] is for applications running in a cloud datacenter, equivalent embedded and mobile ASICs are expected to emerge in the next 1-2 years with the capability to operate multiple, parallel CNN-networks with equivalent levels of power efficiency to the TPU.

C. Contributions of this Work

From the perspective of developing a deep learning model for gaze estimation, the task can either be considered as a regression task or a classification task. Although both are useful, regression provides the greatest predictive flexibility and thus this paper treats the eye gaze estimation task as a regression problem with the goal of finding a gaze angle (ϕ,θ) that corresponds with a low resolution eye image such as one taken from a distance with a simple RGB webcam mounted on a dashboard.

In this paper, a hardware optimized network is implemented with demonstrated suitability for deployment on such consumer devices in terms of memory requirements and speed. This network achieves superior accuracy using a dual channel input technique when compared to other state-of-the-art CNN-based gaze tracking methods for unconstrained, low resolution eye tracking.

II. Related Work

In this section a review of conventional gaze tracking techniques, studies on using low resolution data, and the application of deep learning in gaze estimation are discussed.

A. Contemporary Methods for Eye Gaze Estimation

Gaze tracking algorithms can be broadly classified into model-based methods and appearance-based methods [8]. Appearance-based methods operate directly on the eye images. Model-based methods include 2D and 3D models that use near infrared (NIR) illumination to create corneal reflections to estimate the gaze vector.

Contemporary research on gaze tracking measures accuracy in a wide variety of ways [9], [10], [11], [12]. It is the view of the authors that angular resolution is most reliable and in this work it is used as the metric of accuracy for the proposed algorithm and results are only compared directly with other works that employ the same metric.

These require polynomial or geometric approximations of the human eye to obtain the gaze direction or the point of gaze. Appearance-based methods use eye region images to extract content information such as local features, shape, and texture of eye regions, to estimate gaze direction.

1) 2D Models: 2D models utilize polynomial transformation functions for mapping the gaze vector (vector between pupil center and corneal glint) to corresponding gaze coordinates on a device screen. A number of related works discuss the use of artificial neural networks to perform this mapping [13], [14], [15]. Early papers on this topic proposed support vector machines (SVM) [16] and non geometric methods [17]. Although the approach proposed in this paper differs, we share a similar goal: head pose invariant gaze tracking without a geometric model while treating the eye gaze estimation task as a 2D regression problem.

2) 3D Models: 3D model-based methods typically use a geometrical model of the human eye to estimate the center of the cornea, and the optical and visual axes of the eye [18], [19], [20], [21]. Gaze coordinates are estimated as points of intersection of the visual axes with the scene. These methods achieve high accuracy (1 degree) but require elaborate system setups and knowledge about geometric relations between system components like LEDs, monitors and cameras.

Although such methods achieve high degrees of accuracy, they are not suitable for the use case investigated in this paper because they require high resolution images of the eye and precise geometric measurements. Such models represent current state-of-the-art in AR/VR systems [22].

3) Appearance-Based Methods: Appearance-based methods utilize cropped eye images of a subject gazing at known locations to generate gaze point coordinates. A variety of machine learning methods, using the eye images as input, have been suggested for this task. For the interested reader there is a detailed discussion of appearence based models by Kar et al [9]. Recently, appearance-based methods implemented using deep learning (DL) and convolutional neural network (CNN) approaches have gained momentum. In contrast the other methods discussed in this subsection, CNN methods, learn the features directly from the data rather than employing a preliminary feature detection step and are described in detail in section C.

B. Gaze Estimation from Low Resolution Images

To facilitate gaze tracking in everyday settings, the use of cheap, compact and easy-to-integrate webcams is common but results in poor gaze estimation accuracy. Low resolution images have strong noise effects [23], and distortions in the eye region contours and features become indistinguishable under varying illumination levels, user distance, and movements.

C. Eye Gaze Estimation Using CNN's

Deep learning (DL) techniques have been successfully used in challenging conditions such as those with variable illumination, unconstrained backgrounds and free head motion. For example George and Routray [24] describe a calibration-free real-time CNN-based framework for gaze classification. Two CNNs, for the left and right eyes, are then trained independently to classify the gaze in seven directions.

Zhang et al. [25] describes a novel appearance-based gaze estimation method in which a CNN utilizes the full face image as input with spatial weights on the feature maps to

suppress or enhance information in different facial regions. It achieves high accuracy and robust performance under varied illumination and extreme head poses. Deng and Zhu [26] achieves head pose invariant, 3D gaze tracking using two separate head pose and eye movement models with two CNNs, connected via a gaze transform layer. Finally, Zhang et al [27] builds a CNN to learn the mapping between 2D head angle, eye image and gaze angle (output) using a small LeNet-inspired CNN. For testing, an extensive database is built with more than 200,000 images under variable illumination levels and eye appearances. This database (called MPII Gaze) is also used in this work and its further details are provided in the next section.

Several of the CNN-based works are specifically targeted towards gaze tracking in consumer/handheld devices [28], [29]. Krafka et al [28] presents a CNN-based real time, calibration-free gaze estimation algorithm. It is trained using a large and diverse dataset of eye images taken under variable lighting, head pose, and backgrounds captured from users through a smartphone app. Inputs to their CNN model include eye and face images. The location of the faces in the images are obtained through a face grid, which is used to infer relative eye and head poses. Park and Kim [29] present a calibration-free method using Deep Belief Networks which classify gaze into a grid of nine gaze locations under various head-poses and viewing directions. Zhang, Yao and Cai [30] developed a nine directional CNN-based gaze classifier for a screen typing application, robust to false detections, blinks, and saccades (rapid, abrupt changes in fixation).

D. Related Work Utilizing the MPII Gaze Dataset

Models for eye gaze estimation often have radically different errors on different datasets and this can make comparisons between claimed accuracies meaningless without knowing the accuracy on a common dataset. In this subsection, works that use the same dataset used in this paper are outlined to facilitate such comparison.

The MPII Gaze dataset [27] is large and challenging, containing images collected under a wide range of realistic scenarios, such as varied illumination levels, eye appearances and head poses. Use of MPII gaze dataset for training and testing gaze estimation algorithms can be found in their own papers [27], [25] and also in works by Wu et al [31], Iyer and Ramasangu [32] and Nie et al [33].

Zhang et al [27], which introduces the MPII Gaze dataset, uses a multimodal CNN for gaze estimation and reports a cross dataset test error of 6.1 degrees. Zhang et al [25] uses full face (instead of eye only and multi-region) images with a CNN and achieves a person independent error of 4.8 degrees on MPII Gaze while being robust to illumination variations and extreme head poses. Wu et al, tracks gaze location using a CNN, and tests cross-subject performance on MPII Gaze in addition to that authors' own dataset [34], [35], [36].

Works by Wood et al [34], [35] and Baltrusaitis [37] describe the utilization of the MPII Gaze dataset for comparing face models and other synthetic datasets.

Wood et al [34] presents a method for synthesizing a large set of eye region images with a generative 3D eye region model. Then a gaze estimation method using the k-Nearest-Neighbour algorithm) is tested on the synthetic data and the MPII Gaze dataset to achieve an error of 9.95 and 9.58 degrees respectively. The same authors describe another method for synthetic, labelled photo-realistic eye region image creation using head scan geometry [35]. The generated dataset, along with MPII gaze, is used to test and compare the accuracy of a CNN based gaze estimation method. The CNNs are then tested on the MPII gaze dataset to achieve an error of 7.8 degrees.

E. Discussion

The previous sections summarized relevant work in the field of eye gaze estimation using regression, 3D models, CNNs, and low quality images. The purpose of the detailed review on these methods is to provide the reader with an understanding of the motivation and significance of the work described in this paper. While many papers have been written on conventional polynomial regression-based techniques for gaze estimation, these typically suffer from inaccuracy resulting from head movements and are not as accurate as 3D model-based methods. 3D model-based methods, on the other hand require complicated calibration, arrangement of the physical space, and intensive model calculations, which are often not suitable for implementation in consumer electronic devices. While CNNs have recently been used for gaze estimation, their accuracy is often insufficient and most implementations require powerful and expensive hardware such as graphic processing units (GPUs) due to large network sizes.

The work described in this paper builds on these techniques to achieve superior accuracy while using consumer grade hardware and inexpensive simple setups, which, as can be seen from the literature review, is an essential but relatively new direction of development in eye gaze research.

III. METHODS

The CNN-based gaze estimation methods in this work were evaluated on state-of-the art graphic processing units using python 2.7 and caffe 1.0 with accuracy and euclidean loss layers modified to calculate angle difference in radians. Person-exclusive, leave-one-out cross-validation was used in all experiments.

In eye gaze tracking literature it is common to use the word "accuracy" and "error" interchangeably and this can sometimes cause confusion. For this reason we use the word "error" in any case where the meaning could be unclear. All angles are reported in degrees. Although most deep learning papers will use just one training, testing, and validation set, such methods are not recommended in cases where there are few individuals in the dataset. For this reason we use a "leave one out" training and testing strategy. Besides being the most appropriate testing method for a dataset with only 15 individuals, it is also necessary for comparison with the other published works that use this dataset as any other training and testing methodology would produce incomparable results. This type of leave-one out cross validation is a special case of k-fold cross validation where the number of folds is the same as the number of distinct entities.

In this paper, error was determined as the average euclidean distance between the ground truth and predicted angles on the left-out, person-exclusive test set as follows:

1) For persons 0 to n (where n is the number of distinct individuals in the dataset):
   a) Train a model on images and labels from all identities except for the one chosen above such that the model outputs predicted gaze directions (ϕ, θ)
   b) Test model using eye images from selected ("left-out") identity.
   c) For each prediction (ϕ, θ) and ground truth ($\hat{\phi}$, $\hat{\theta}$) above calculate the euclidean distance between them.
   d) The mean value of the above distances is the error for this fold.
2) The sum of all the errors is divided by the number of folds to find the mean error – The error reported for a given neural network architecture is the mean error for all the folds.

Multiple deep neural networks were compared for eye gaze estimation. The publicly available MPII Gaze dataset was used for all experiments except for the first, where the UT Multiview dataset is also used.

A. Conventions for CNN Diagrams and Figures

A number of conventions for describing the layers of neural networks have arisen in deep learning literature. The authors of this paper choose to follow a common variant of these conventions. In general, the convention is that the type of layer is followed by the number of output parameters in parentheses. For example Conv(40) would indicate a convolutional layer with 40 output layers. When it is necessary to know the kernel size, this is also indicated by an @ symbol followed by the kernel size. For example, Conv(40@3x3) would indicate a convolutional layer with 40 outputs and a 3x3 kernel size.

When a layer type is followed immediately by a number, this number indicates the relative position of the layer compared to previous and subsequent layers of the same type. For example, conv1 indicates that this is the first convolutional layer, whereas conv2 would be the second convolutional layer.

The input layer is a special type of layer that describes how raw data is organized. In the case of this paper, the input layers take either eye crops or head poses. For input layers that take eye crops, the following format is used: Input([number of channels]x[image width]x[image height]).

Head pose inputs are not images but are instead simply angle values represented as floating point numbers. Input(ϕ,θ) means that the phi and theta values are being passed to the network at that point in the figure.

In the case of pooling operations the output size is determined by the input size divided by the size of the pool parameter. Max Pool(2x2) would indicate a 2x2 max pooling operation on the previous layer.

RELU is defined as max(0,x) where x is the input. In all cases the size of the output of a relu layer is equal to the size of the output of the preceding layer.

Dropout layers are a regularization technique that helps prevent over-fitting and also increases the resiliency of a network. It is only used during training. Dropout randomly sets a percent of the weights in the network to zero. For example, dropout(0.5) indicates that 50% of the weights in the preceding layer should be set to zero in a random fashion. The output of dropout is always the same size and shape as the layer on which it operates.

FC or fc indicates a fully connected layer and fc(100) would indicate a fully connected layer with 100 outputs. In such layers, every "neuron" or unit is connected to every neuron of the previous layer. It is common to use such a layer after all the convolutional layers. For more detailed information about these layers, the reader is encouraged to consult the reference section, particularly [38], which contains a detailed description of the common types of layers used in modern deep learning approaches.

B. MPII Gaze Dataset Details

The MPII gaze dataset is a large collection of 213,659 images captured under unconstrained conditions from 15 subjects over several days. The images are collected under multiple illumination conditions. Some of the subjects wear spectacles and some do not. The images were captured at various gaze angles, recorded by software running on the participant's laptops. In each session, the subjects were asked to look at random sequences of 20 onscreen positions and to confirm their attentiveness, the subjects were asked to press the space bar once the onscreen target was disappearing.

The dataset contains eye and head features and target (gaze angle) values for every participant. To use MPII Gaze, the authors suggest mapping their reported vector to angles using a Rodrigues transformation, and this has been done for all reported experiments. The Rodrigues transformation is given by the Rodrigues rotation formula which provides a rotation matrix from which roll, pitch, and yaw can be generated [39].

IV. Experiments and Results

In this section, multiple experiments are described to provide insight on 4 primary research questions. These are tested on multiple CNN architectures and are discussed in this section. One of the first research goals was to achieve state of the art test error on a network that could perform inference within 3-15 ms on a typical single proprietary low power consumer embedded device.

The specific research questions are:

1) How does an architecture that uses both eyes compare to one that uses one eye in terms of accuracy?
2) How does simulated camera distance impact eye gaze accuracy for the proposed model?
3) Can augmentation be used to reduce any negative impacts?
4) Can the proposed hardware-friendly architecture perform with sufficient accuracy and speed?

First, the intra-dataset, person-exclusive experiments from Zhang et al [27] were duplicated. The same procedure to estimate accuracy was used except the altered accuracy layers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2899869, IEEE Transactions on Consumer Electronics

were modified to eliminate not a number (NaN) errors by replacing undefined values of the arc cosine function with the largest or smallest valid values, as appropriate.

A. Approach 1: Analysis of Eye Flipping

In Zhang et al [27], one network is used for both eyes (with one inference per eye) and one of the eyes is flipped so that the gaze angle is roughly correct. An experiment was designed to see if this flipping had an impact on model accuracy. Six experiments were performed using the UT and MPII-Gaze datasets to see if training on both eyes or just one eye impacted accuracy. By doing experiments with combined and non combined datasets, it was also possible to determine weather they had similar distributions, and thus, if combining the two would be helpful for future experiments.

TABLE I:
RESULTS OF APPROACH 1

Training	Error
MPII Left eye only	5.9 degrees
MPII+UT Left eye only	5.6 degrees
MPII Both eyes	7.4 degrees
MPII+UT Both eyes	6.5 degrees
MPII Right eye only	5.3 degrees
MPII+UT Right eye only	6.1 degrees

Analysis of the errors due to choice of input during training (left, right, or flipped (both)). Training a network on both eyes using the flipping approach suggested in [27] appears to be a source of error. This observation informs the remaining experiments in the paper, which use a 2 channel approach instead.

As shown in Table I, the method of individually classifying eye images and simply adjusting the right eye and angles as used in Zhang et al [27] is a limiting factor in accuracy for that method. In both datasets, the performance was increased exclusively using left eyes or right eyes. This suggests that simply flipping the eye as suggested by Zhang et al [27] may be a source of error in their model.

These results also indicated that the distribution of MPII Gaze and UT-Multiview are sufficiently different that combining the two for training gives no, or very little, improvement. Because of this, it was decided to use only MPII Gaze for the remaining experiments in this section as UT-Multiview had no significant influence on error.

B. A New Approach: Dual Eye Channels

Given the problems identified in the previous subsection with flipping one of the eyes, and not wanting to use two different networks for reasons of efficiency, a new approach involving using both the left and right eyes in separate input channels was investigated. Specifically, the left eye and right eye images are passed to the network in channels 0 and 1 respectively, and the gaze and pose information are averaged between the left and right eye images to create a single gaze and pose vector. Due to the results in the previous section, which indicated that data from UT did not significantly impact the results, only the MPII Gaze dataset was used.

This modified, two channel architecture resulted in a significant increase in accuracy, averaging 4.63 degrees of error between the target and predicted values on unseen individuals. A diagram of this network can be seen in Fig 1.

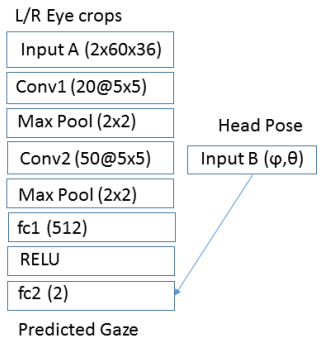


Fig. 1: Diagram of dual eye channel CNN used for eye gaze estimation. In contrast to previously published approaches, the right and left eye crops are input to the network simultaneously along with a single head pose vector. The output of the network is a single predicted pitch and yaw angle for both eyes, as if it were a single eye located exactly between the right and left eyes looking at the same object. This network was used for the experiment in sub-section V B and it was found to increase accuracy over previously published models. Please see section III.A for a description of what the text in each of these layers means.

C. Efficiency: Can We Reduce the Number of Parameters?

Deep neural networks can often be made more efficient by reducing the number of parameters but this can sometimes come at the cost of accuracy. To see if reducing the number of parameters was possible without harming accuracy, an experiment was performed to halve the size of all output parameters. This experiment was not allowed to run for the full duration because the exact angle accuracy did not matter, only evidence that the network complexity could be reduced to a point where it would be small enough if necessary. This resulted in an average error of 4.980% on an unseen individual from the MPII Gaze dataset and indicates that reducing the number of parameters had little impact on accuracy.

D. Multi Resolution Experiments

Eye gaze systems in consumer devices must be able to maintain accuracy at a large range of distances. Although MPII Gaze has some variability in distance from the camera, the distances are not realistic for the conditions expected in, for example, a driver monitoring system or a distant cell phone camera. An experiment was conducted to determine if the network was able to perform under an expectedly wide range of subject distances.

Specifically, the goal was to accommodate realistic distances between the camera and the subject in situations that would be typical in commercial eye trackers that utilize low cost, low resolution cameras. To simulate the loss of information caused by distance, down-sampling using nearest neighbor interpolation was performed on the eye images in MPII Gaze as follows:

- Input image 60 x 36 ->Downscale to 52 x 31 ->Upscale to 60 x 36 ->CNN Eye gaze angle.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCE.2019.2899869, IEEE Transactions on Consumer Electronics

- Input image 60 x 36 ->Downscale to 26 x 16 ->Upscale to 60 x 36 ->CNN Eye gaze angle.

As can be seen in Table II, the network learned a narrow range of distances, and performance deteriorates when the subject is further from the camera than those in the training set. As a sanity check, an experiment was done to see if the downscaling algorithm was at fault for the poor results, so in addition to nearest, we also tried bicubic, linear, and LANCZOS using open source computer vision software. The experiment showed that the downscaling algorithm used had no influence on the results.

TABLE II:
RESULT OF DISTANCE SIMULATION AND AUGMENTATION EXPERIMENTS

Resolution	Unaugmented error	Augmented error
60 x 36	4.63 degrees	4.918
52 x 31	9.90 degrees	4.94
26 x 16	10.10 degrees	4.98

This table shows the impact of camera distance and augmentation for a trained eye gaze model on angle accuracy using the network from experiment 2.

This demonstrates that the model is sensitive to changes in distance. In the next section, an experiment is performed to see if data augmentation can be used to improve upon this.

E. Impact of Random Resizing as Augmentation

Data augmentation has been shown in many studies [40] to have a large impact on model performance but augmenting to increase accuracy of on a wide range of distances appears to be neglected in literature on eye gaze. To further improve accuracy, the dataset was augmented with multiple randomly chosen resolutions to match the full range of desired distances. To help reduce the chance that the network would learn the specific interpolation method used, Nearest is used in the training set, but Lanczos filtering is used in the testing set.

These results indicate that augmenting the images with distances that are likely to be encountered in real world usage situations is an effective way to increase accuracy and succeeds in achieving some invariance to subject distance.

F. A Quest for Hardware Efficiency and Even Better Accuracy

It was shown by Simonyan and Zisserman [41] that two stacked layers of 3x3 convolutions have the same receptive field as a single 5x5 layer, with fewer multiplications. Because the efficiency of a convolutional neural network is primarily dependent on the number of multiplications, and thus the number of convolutions, stacked 3x3 convolutions were chosen for this experiment and the network was redesigned accordingly. Several experiments involving architectures with stacked 3x3 kernels were performed using different parameters. The best two architectures were further evaluated, and the best models from each of them were chosen and evaluated on multiple resolutions as shown in Table III. A diagram of the final architecture used can be seen in fig 2, and a comparison with other published works can be seen in Table IV.

In order to maximize efficiency on a particular proprietary DSP, it was necessary to alter the output sizes of this network to be powers of 2. This is the explanation for why the output sizes are increased for this network compared to the others evaluated previously.

TABLE III:
ERROR OF PROPOSED MODEL FOR VARIOUS RESOLUTIONS (DEGREES)

Model	36 x 60	31 x 52	26 x 16
Best	3.650	4.1690	4.240
Second best	4.100	4.324	4.366
Benchmark	4.917	4.940	4.970

The benchmark method is the best performing network from experiment 2, trained and tested with the same techniques. The second best model is called Network 4 in subsection H of this section and the best model is network 5.

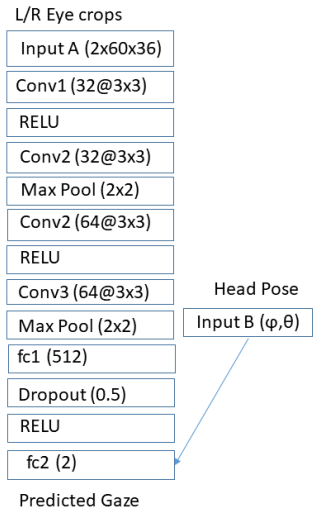


Fig. 2: This figure contains a diagram of the proposed network. It was found to further improve error over the method developed in subsection B from 4.63 degrees to 3.65 degrees. Like the network in fig 1, this network takes both right and left eye image crops as input, using 2 channels and uses the same method to merge the left and right eye gaze vectors for the ground truth data. Another difference is the stacked 3x3 convolutions replacing the 5x5 convolutions and the addition of RELU nonlinearities The error of this network is compared with the error from other experiments in this paper in Table III. Please see section III.A for a description of what the text in each of these layers means.

G. Runtime analysis of the used networks

In this section, the execution time of the 5 networks used in this paper are evaluated on commodity processors. The result of these experiments are shown in Table IV.

Network 1 is the "one channel" approach used in [27] which we compare with. In contrast to the new approaches, this method performs a separate inference for each eye and flips the right eye.

Network 2 is the two channel improvement discussed in experiment 2 and shown in figure 1.

Network 3 is the variation of network 2 with half as many outputs per layer which was used in experiment 3.

Network 4 is a version of network 2 with 3x3 kernels but with the same number of units per layer as network 2. It is included to allow the reader to differentiate runtime that resulted from increasing the number of outputs from runtime that resulted from the change in network architecture.

Network 5 is the proposed model that has the best accuracy and utilizes the 3x3 kernels instead of 5x5 kernels. A diagram of this network is shown in figure 2.

The embedded processor is a typical 64 bit processor used in embedded and mobile systems, and available on a well-known embedded prototyping platform. Tests used only one core of this processor. The second processor is a popular high end workstation central processing unit (CPU). For a fair comparison, only one thread was used for tests. Lastly the GPU used is a popular high end consumer GPU targeted at video gamers. As can be seen in Table IV, network 3 is significantly faster than the other networks while network 5 provides a good balance between speed and accuracy. In some cases network 3 may be preferred due to increased speed and competitive accuracy.

TABLE IV:
FRAMES PER SECOND OF CNN ON COMMODITY HARDWARE

Model	Embedded	CPU	GPU
Network 1 [27]	20.64	473.11	3984.09
Network 2	39.81	960.06	8078.47
Network 3	92.59	3080.80	12607.90
Network 4	29.68	592.02	6134.02
Network 5	19.11	400.50	5500.37
VGG16 [41]	0.0043	0.64	120.4

All units are in frames per second. Details of these networks are discussed in subsection H. As can be seen from this table, networks 2-4 perform faster on all tested hardware than the comparison networks (Network 1 and VGG16) while providing greater accuracy. Network 4 has similar performance to network 1 while increasing accuracy. The well known VGG16 network is included to allow the reader to see the speed gain over this type of architecture. VGG16 is an improvement over Alexnet, which was used to achieve 4.8 percent accuracy on the same dataset [25].

H. Contributions

When deploying eye gaze solutions for consumer devices there are two important aspects to consider: Accuracy and efficiency. This paper contributes to addressing both issues by demonstrating improved accuracy and also by reducing the number of multiplications needed for predictions, thus increasing efficiency. It should be noted that the total number of matrix multiplications needed to obtain predictions from a convolutional neural network is determined by the size of the convolutional kernels, the step size, the number of nodes in each layer, and the number of layers [38]. These multiplications are often measured in multiply-accumulate operations (MACs).

For a digital signal processor (DSP) that primarily performs deep learning inference, the power consumption requirements are directly related to the number of MACs required per inference [43] [44] [45].

TABLE V:
COMPARISON OF PROPOSED MODEL WITH OTHER PUBLISHED WORKS ON THE MPII-GAZE DATABASE

Citation	Error (degrees)
Baltrusaitis et al [36]	9.96
Wood et al [34]	9.58
Shrivastava et al [42]	7.8
Nie et al [33]	7.1
Zhang et al [27]	6.1
Zhang et al [25]	4.8
Proposed	3.65

This table shows the best performing network from this paper (Proposed) compared with other reported results on the MPII-Gaze dataset.

A variant of the proposed algorithm is now operating effectively on a hardware CNN SoC (system-on-chip) engine with 32 megabytes of random access memory, demonstrating that this approach is viable for low-resource consumer electronic devices.

At the beginning of this section we followed Zhang et al's approach [27] in subsection A. In subsection B, we found that by changing the network architecture to accept two eye crops, one for the left and one for the right eye in two input channels, and merging the gaze vectors and the position vectors, we were able to improve accuracy over that reported in Zhang et al [27]. It was then shown in subsection C that the architecture introduced in subsection B could be made more efficient by halving the number of nodes in each layer with a negotiable decrease in accuracy.

Subsection D demonstrated that the error more than doubles when the network is exposed to images that have been artificially resized to simulate a wide range of distances between the subject and the camera. This problem had not been addressed in previous work. It was then shown in subsection E that this increase in error could be nearly eliminated by use of data augmentation. Finally, the results of the experiment conducted in subsection F suggest a new network architecture that outperforms previously published works while reducing the number of multiplications and thus increasing efficiency.

V. CONCLUSION

Our results show that using information from both eyes in the neural network can increase accuracy. This is demonstrated in section V, where adding additional eye information from the opposite eye enabled improved results over individual eyes, helping the network make sense of low quality images with ambiguous gaze. As expected, in all cases, the deeper network had the best performance. This research demonstrated the sensitivity of such models to variations in distance and how data augmentation can be used to overcome this. Most importantly, a new compact hardware-friendly architecture designed for use in small consumer electronics has been introduced and evaluated on the eye gaze task.

When evaluated on MPII Gaze, the proposed model performs favorably (see Tables IV and V) even when compared with much larger networks in the literature.

Running an optimized CNN based algorithm, as described in this paper can provide a high-performance, low-energy solution for continuous eye-tracking in next generation consumer

electronic products such as ultra-lightweight smartphones and augmented reality glasses.

Since augmentation resulted in a significant improvement in accuracy, it may be fruitful to try other types of augmentation such as Generative Adversarial Networks (GANS) with landmarks, [46] and Smart Augmentation (SA) [40] in future work. This will either require modifying such methods to work on regression problems, or translating the eye gaze problem into a classification task for the purpose of generating augmented data and then back to a regression task. Additionally, there are plans to investigate whether temporal information [47] can be used to further increase the accuracy without sacrificing the need for performance, as it has been shown to increase performance in DMS systems.

REFERENCES

[1] Y. Liang, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 340–350, June 2007.

[2] E. Wood and A. Bulling, "Eyetab: Model-based gaze estimation on unmodified tablet computers," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, ser. ETRA '14. New York, NY, USA: ACM, 2014, pp. 207–210. [Online]. Available: http://doi.acm.org/10.1145/2578153.2578185

[3] A. Tawari and M. M. Trivedi, "Robust and continuous estimation of driver gaze zone by dynamic analysis of multiple face videos," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, June 2014, pp. 344–349.

[4] V. Vaitukaitis and A. Bulling, "Eye gesture recognition on portable devices," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, ser. UbiComp '12. New York, NY, USA: ACM, 2012, pp. 711–714. [Online]. Available: http://doi.acm.org/10.1145/2370216.2370370

[5] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput*, vol. 1, no. 4, pp. 541–551, 1989.

[6] J. Lemley, S. Bazrafkan, and P. Corcoran, "Deep learning for consumer devices and services: Pushing the limits for machine learning, artificial intelligence, and computer vision." *IEEE Consum. Electron. Mag.*, vol. 6, no. 2, pp. 48–56, 2017.

[7] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Computer Architecture (ISCA), 2017 ACM/IEEE 44th Annual International Symposium on*. IEEE, 2017, pp. 1–12.

[8] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, 2010.

[9] A. Kar and P. Corcoran, "A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms," *IEEE Access*, vol. 5, pp. 16 495–16 519, 2017.

[10] F. L. Coutinho and C. H. Morimoto, "Augmenting the robustness of cross-ratio gaze tracking methods to head movement," in *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 2012, pp. 59–66.

[11] S. Chen and C. Liu, "Eye detection using discriminatory haar features and a new efficient svm," *Image Vision Comput.*, vol. 33, pp. 68–77, 2015.

[12] H. chuan Lu, C. Wang, and Y. w. Chen, "Gaze tracking by binocular vision and lbp features," in *2008 19th International Conference on Pattern Recognition*, Dec 2008, pp. 1–4.

[13] Z. Zhu and Q. Ji, "Eye and gaze tracking for interactive graphic display," *Machine Vision and Applications*, vol. 15, no. 3, pp. 139–148, Jul 2004. [Online]. Available: https://doi.org/10.1007/s00138-004-0139-4

[14] C. Jian-nan, Z. Chuang, Y. Yan-tao, L. Yang, and Z. Han, "Eye gaze calculation based on nonlinear polynomial and generalized regression neural network," in *2009 Fifth International Conference on Natural Computation*, vol. 3, Aug 2009, pp. 617–623.

[15] J. Wang, G. Zhang, and J. Shi, "2d gaze estimation based on pupil-glint vector using an artificial neural network," *Applied Sciences*, vol. 6, no. 6, 2016. [Online]. Available: http://www.mdpi.com/2076-3417/6/6/174

[16] Z. Zhu, Q. Ji, and K. P. Bennett, "Nonlinear eye gaze mapping function estimation via support vector regression," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 1, 2006, pp. 1132–1135.

[17] J. Zhu and J. Yang, "Subpixel eye gaze tracking," in *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, May 2002, pp. 124–129.

[18] E. D. Guestrin and M. Eizenman, "General theory of remote gaze estimation using the pupil center and corneal reflections," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1124–1133, June 2006.

[19] T. Ohno and N. Mukawa, "A free-head, simple calibration, gaze tracking system that enables gaze-based interaction," in *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications*, ser. ETRA '04. New York, NY, USA: ACM, 2004, pp. 115–122. [Online]. Available: http://doi.acm.org/10.1145/968363.968387

[20] Z. Zhu and Q. Ji, "Novel eye gaze tracking techniques under natural head movement," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 12, pp. 2246–2260, Dec 2007.

[21] D. Beymer and M. Flickner, "Eye gaze tracking using an active stereo head," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2, June 2003, pp. II–451–8 vol.2.

[22] J. Lemley, A. Kar, and P. Corcoran, "Eye tracking in augmented spaces: a deep learning approach," in *2018 IEEE Games, Entertainment, Media Conference (GEM)*. IEEE, 2018, pp. 385–390.

[23] Y. Fu, W. P. Zhu, and D. Massicotte, "A gaze tracking scheme with low resolution image," in *2013 IEEE 11th International New Circuits and Systems Conference (NEWCAS)*, June 2013, pp. 1–4.

[24] A. George and A. Routray, "Real-time eye gaze direction classification using convolutional neural network," in *2016 International Conference on Signal Processing and Communications (SPCOM)*, June 2016, pp. 1–5.

[25] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "It's written all over your face: Full-face appearance-based gaze estimation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 2299–2308.

[26] H. Deng and W. Zhu, "Monocular free-head 3d gaze tracking with deep learning and geometry constraints," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 3162–3171.

[27] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 4511–4520.

[28] K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba, "Eye tracking for everyone," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 2176–2184.

[29] H. Park and D. Kim, "Gaze classification on a mobile device by using deep belief networks," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Nov 2015, pp. 685–689.

[30] C. Zhang, R. Yao, and J. Cai, "Efficient eye typing with 9-direction gaze estimation," *Multimedia Tools and Applications*, Nov 2017. [Online]. Available: https://doi.org/10.1007/s11042-017-5426-y

[31] X. Wu, J. Li, Q. Wu, and J. Sun, "Appearance-based gaze block estimation via cnn classification," in *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*, Oct 2017, pp. 1–5.

[32] S. D. Iyer and H. Ramasangu, "Hybrid lasso and neural network estimator for gaze estimation," in *2016 IEEE Region 10 Conference (TENCON)*, Nov 2016, pp. 2579–2582.

[33] S. Nie, M. Zheng, and Q. Ji, "The deep regression bayesian network and its applications: Probabilistic deep learning for computer vision," *IEEE Signal. Proc. Mag.*, vol. 35, no. 1, pp. 101–111, Jan 2018.

[34] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling, "Learning an appearance-based gaze estimator from one million synthesised images," in *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ser. ETRA '16. New York, NY, USA: ACM, 2016, pp. 131–138. [Online]. Available: http://doi.acm.org/10.1145/2857491.2857492

[35] E. Wood, T. Baltruaitis, X. Zhang, Y. Sugano, P. Robinson, and A. Bulling, "Rendering of eyes for eye-shape registration and gaze estimation," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 3756–3764. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2015.428

[36] T. Baltrusaitis, P. Robinson, and L. P. Morency, "Openface: An open source facial behavior analysis toolkit," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2016, pp. 1–10.

[37] U. Weidenbacher, G. Layher, P. M. Strauss, and H. Neumann, "A comprehensive head pose and gaze database," in *2007 3rd IET International Conference on Intelligent Environments*, Sept 2007, pp. 455–458.
[38] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.
[39] E. Trucco and A. Verri, *Introductory techniques for 3-D computer vision*. Prentice Hall Englewood Cliffs, 1998, vol. 201.
[40] J. Lemley, S. Bazrafkan, and P. Corcoran, "Smart augmentation learning an optimal data augmentation strategy," *IEEE Access*, vol. 5, pp. 5858–5869, 2017. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2696121
[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[42] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 2242–2251.
[43] C. Turner, "Calculation of tms320lc54x power dissipation," *Technical Application Report SPRA164, Texas Instruments*, 1997.
[44] J. Lemley, S. Bazrafkan, and P. Corcoran, "Learning data augmentation for consumer devices and services," in *Consumer Electronics (ICCE), 2018 IEEE International Conference on*. IEEE, 2018, pp. 1–3.
[45] A. Abnous, K. Seno, Y. Ichikawa, M. Wan, and J. Rabaey, "Evaluation of a low-power reconfigurable dsp architecture," in *International Parallel Processing Symposium*. Springer, 1998, pp. 55–60.
[46] S. Bazrafkan, H. Javidnia, and P. Corcoran, "Face synthesis with landmark points from generative adversarial networks and inverse latent space mapping," *arXiv preprint arXiv:1802.00390*, 2018.
[47] J. Lemley, S. Bazrafkan, and P. Corcoran, "Transfer learning of temporal information for driver action classification," in *The 28th Modern Artificial Intelligence and Cognitive Science Conference (MAICS)*, 2017.

Joseph Lemley (S'16) received a B.S. degree in computer science from Central Washington University in Ellensburg Washington, USA in 2006. He earned a M.S in computational science from Central Washington University in Ellensburg Washington, USA in 2016. He is currently pursuing the Ph.D. degree in electronic engineering at the National University of Ireland, Galway.

He is a research and development engineer at Fotonation in Galway, Ireland, under the IRCSET Employment Ph.D. Program. His research interests include Artificial Intelligence, Deep Learning, and Computer Vision.

Mr. Lemley received the 2017 Best paper joint award for IEEE Consumer Electronics Magazine, best paper second place award at ICCE 2018 and other awards during previous years.

Anuradha Kar (S'08) was born in Kolkata, India and received the Bachelor of Technology degree in electronics & communication engineering from the West Bengal University of Technology, Kolkata, India, in 2009 and Master of Technology degree in radio, physics and electronics from University of Calcutta, in Kolkata, India in 2011. She is currently pursuing the Ph.D. degree with the National University of Ireland, Galway.

Her research interests include human computer interaction, application of eye gaze in next generation consumer applications like AR/VR and smart devices, and performance evaluation of sensor systems.

Ms. Kar won second position at the All India IEEE Student project contest in 2009, and was awarded a two year Master degree fellowship from the Indian Space Research Organization (ISRO) between 2009-2011.

Alexandru Drimbarean (M'16) received his B.S in electronic engineering from Transilvania University of Brasov, Romania in 1997 followed by the M.Sc. in electronic engineering at National University of Ireland, Galway in 2002.

He has been the Vice President of Advanced Research at FotoNation in Galway, Ireland since 2015 and has worked with Fotonation in various research, engineering, and management roles since Jun 2000. He has authored several journal articles as well as more than 30 patents. His interests include image processing, and understanding as well as computer vision and machine learning.

Peter Corcoran (M'95-F'10) received the BAI (electronic engineering) and BA (maths) degrees from Trinity College Dublin, Ireland in 1984. He continued his studies at TCD and was awarded the Ph.D. in electronic engineering in 1987 for research work in the field of Dielectric Liquids.

He is a professor with the Department of Electrical and Electronic Engineering at the National University of Ireland, Galway where he has taught since being appointed to a lectureship in 1986. He is co-author on 350+ technical publications and co-inventor on more than 300 granted US patents. His research interests include biometrics, imaging, deep learning, edge-AI and consumer electronics.

Professor Corcoran has received numerous industry and academic awards and honors including being elected as an IEEE Fellow in 2010. He is the past Editor-in-Chief of IEEE Consumer Electronics Magazine.

Appendix J: Eye Tracking in Augmented Spaces: A Deep Learning Approach

Eye Tracking in Augmented Spaces: a Deep Learning Approach

Joseph LEMLEY, Anuradha KAR and Peter CORCORAN, *Fellow, IEEE*
Center for Cognitive, Connected & Computational Imaging, College of Engineering & Informatics,
NUI Galway, Galway, Ireland
Email: { j.lemley2, a.kar2, peter.corcoran}@nuigalway.ie

***Abstract*—The use of deep learning for estimating eye gaze in augmented spaces is investigated in this work. There are two primary ways of interacting with augmented spaces. The first involves the use of AR/VR systems; the second involves devices that respond to the user's gaze directly. This domain can overlap with AR/VR environments but is not exclusive to them and contains its own unique set of issues. Deep learning methods for eye tracking that are capable of performing with minimal power consumption are investigated for both problems.**

Keywords: Augmented reality, Virtual reality, gaze estimation, deep learning, convolutional neural networks, smart spaces

I. INTRODUCTION

Eye gaze estimation is one of the most challenging frontiers of deep learning (DL) research in vision tasks and one of the few areas where conventional approaches are still dominant [1].

Recently, deep learning has been able to surpass conventional approaches in difficult gaze estimation tasks such as in DMS systems or handheld devices [2][3]. The strength of DL lies in inferring gaze, or performing subtasks in more complex gaze detection systems, even with poor image quality. Further, with novel data augmentation techniques as [4], DL has prospects to achieve good classification abilities using smaller network sizes and hardware-friendly architectures, which make it attractive for use in consumer electronic devices.

An augmented spaces is defined as any physical space with additional sensory or display elements that augment the physical function, utility, and purpose of that space. This is achieved through the use of sensors, actuators, computing and networking devices incorporated within the space that enable recognition of humans, their activities and gestures in real time [5][6]. Examples may include a television that reduces power to screen when no one is looking at it, automatic room brightness adaptation based on human presence or motion sensor based security systems [7] .

With respect to eye tracking in augmented spaces, there are the two types of configurations to consider when designing gaze based interactions. The first type involves smart glasses or head mounted AR/VR systems. In these, the gaze tracking task is equivalent to that in conventional AR/VR applications where high quality close-up images of the eye are used and the subject interacts with nearby objects at least partly through the AR/VR system. The second scenario involves individual smart devices and appliances each with their own camera systems that can respond to gaze. If no one is watching the TV or digital readout, why should it operate at full power? Useful information and controls can be displayed only when someone is actively paying attention to them. These systems typically need to track the eye movements of everyone in their vicinity, in real time and with varying lighting conditions and user distances. Gaming is another domain which benefit from gaze information[8]. For example, if a player is momentarily distracted and not looking at the screen, it would be a bad time to introduce a crucial plot element or battle. Gaze information can also be used to provide an element of surprise in dynamic scenes by spawning enemy NPC (Non player characters) away from the players current gaze. However, the distance between the eye tracker camera and the player, and player head movements are significant factors determining accurate gaze tracking in gaming applications. AR/VR gaming systems typically have close eye facing cameras and are able to capture detailed eye images, whereas gaze tracking for games on remote game consoles, cell phones and other devices typically have cameras at a distance and varying user head movements, making eye tracking challenging for them.

In this paper we examine the use of deep learning for gaze tracking in AR/VR as well as remote setups for augmented environments. In the first case, DL is still behind conventional methods which can achieve accuracies as high as 0.5 degrees [9]. In the second case however, DL is emerging as the most feasible solution due to its ability to make sense of eye images that are too low quality for conventional approaches[10].

The paper is organized as follows: in section II.A, eye gaze estimation in augmented and virtual reality (AR & VR respectively) applications is explained. Gaze information has a significant impact in AR/VR environments [11][12], where gaze directions and gaze based functions are used to make user experiences more immersive, natural and effortless. In section II B gaze estimation in pure augmented spaces, without the use of AR/VR systems is explained. These include the tracking and application of gaze information for domotic controls, multimedia communications or assisted living systems [13].

The use of deep learning for gaze estimation in augmented spaces is primarily an unexplored area of research and dedicated network models or datasets for gaze estimation in such environments are not publicly available. Therefore in Section III, we describe our methodologies of using convolutional neural networks (CNN) and two different gaze datasets- one captured using a head-mounted eye tracker and another with normal remote setup to test our algorithm performance. The description of the CNN model, two datasets and proposed approaches pertaining to the two types of eye gaze estimation methods for augmented spaces are presented, followed by discussions and conclusion in Section IV.

978-1-5386-6304-2/18/$31.00 ©2018 IEEE

II. EYE TRACKING IN AUGMENTED SPACES: PROSPECTS & CHALLENGES

A. Eye tracking in AR/VR applications

The significance of tracking user eyes in an AR/VR environment has been recently acknowledged in literature. In Augmented Reality applications, gaze information is fused with data from a scene camera to estimate the point of gaze of a user, for applications such as reading and document retrieval as described in [9]. In this work, eye tracking is used to identify the part of a document the user is reading and display relevant information on the see-through head mounted display (HMD). Gaze information, coupled with other eye movement features like dwell time and blinks can be used for object selection [12] and zooming and capturing snapshots in AR devices [14]. Gaze has also been used for wearable context aware messaging service in [15] and for attention guidance using peripheral vision in AR headsets in [16].

In Virtual reality (VR) research, eye movements can be used for interaction such as in [17]. [18] presents an immersive 3D VR interface with gaze based interactions such as menu selection and gaze directed typing of mails using a virtual keyboard. Realistic rendering of characters and social interaction between a user and virtual characters may be established by combining dynamic facial appearances and gaze directions as described in [19]. Other purposes of using gaze in VR include achieving wide view panoramas, foveated rendering and natural exploration of virtual environment [20].

B. Eye gaze tracking in non-AR/VR augmented spaces

Intelligent environmental controls using eye gestures fall into this category. In the consumer device domain, there has been developments of gaze controlled TV [21] which senses gaze to enable screen brightness variations, menu selection and understanding user program preferences. For assisted living applications, gaze based control of wheelchairs[22] and home appliances have been proposed[23], and also eye tracking as diagnostic technology for patients with disabilities[24].

TABLE I. SUMMARY OF SOME RECENT WORKS USING EYE GAZE IN AUGMENTED SPACES

Citation	Type	Setup	Applications	Gaze accuracy
[9]	AR/VR	HMD and eye tracker	Assisting reading activity	0.5 degrees
[14]	AR/VR	HMD with eye tracker and lenses	Eye directed zooming	0.5 degrees
[17]	AR/VR	HMD with eye tracker	Gaze based object selection	0.6 degrees
[18]	AR/VR	HMD with head and eye trackers	Gaze based multimedia use	1 degree
[19]	AR/VR	3D Stereo Rig with remote eye tracker	Gaze-aware facial re-enactment in VR	1.5 degrees
[21]	Non-AR/VR	Remote tracker with camera, LED, lens	TV display input and control	1.32 degrees
[22]	Non-AR/VR	Remote tracker	Wheelchair motion control with gaze	0.5 degrees
[23]	Non-AR/VR	Camera	Appliances detect and respond to gaze	--

C. Challenges of eye tracking in augmented spaces

Eye tracking in AR/VR headsets may face significant and unique challenges as described in [25]. These include motion and depth of field blur, latency, rapid calibration drifts while following smooth pursuit eye movements and loss in tracking due to varying orientation of the eye camera with respect to the eye. Complicated system design, bulky processing units and high power consumption may also limit the usability of these devices in the consumer electronics domain.

Major problems in using remote gaze trackers for estimating gaze as an input modality arise from the issue of Midas touch [26], difficulty in handling complex tasks by gaze gestures and expensive hardware.

III. DEEP LEARNING FOR GAZE ESTIMATION IN AUGMENTED SPACES.

A. Concept and approach

Deep learning has been successful in achieving good accuracy in remote gaze estimation problems [27]. However, implementing DL based eye tracking in AR/VR is difficult due to lack of publicly available datasets built for gaze tracking in AR/VR environments. In this work, gaze estimation for both the eye tracking scenarios in an augmented space is approached. For the AR/VR problem, we start by training a DL model with eye images captured using a head-mounted eye tracker. We then progressively introduce complex variations in the images typical to those faced by eye trackers in AR/VR environments. For the non-AR/VR eye tracking condition, we use a low resolution gaze dataset, which typically has the characteristics of images captured by remote eye trackers.

B. Eye datasets used

Since augmented spaces can be experienced through either AR/VR or through smart devices that are farther from the users eyes, it is necessary to use separate datasets to train and test the two DL methods. In this section we describe two gaze datasets: a high resolution dataset (for AR/VR) and low resolution datasets for gaze tracking on smart devices having their own cameras.

1) High resolution datasets for AR/VR

There is a dearth of public datasets from which deep learning systems can be trained (or evaluated) for eye gaze estimation utilizing head mounted eye-facing cameras. The only suitable publically available dataset found is the one developed by McMurrough et al called the "Point of Gaze (PoG) Eye Tracking Dataset"[28]. Unfortunately, this dataset only has images of the right eye and therefore may not be used for AR applications where knowing what a person is looking at in 3D space involves calculating the intersection of two gaze vectors. This information is still useful because there are typically only a few objects that collide with a given gaze vector that are within a person's field of view and these can all be assumed to be the gaze target in an augmented or virtual space. Despite this limitation, the PoG Eye Tracking Dataset is the most suitable publically available dataset captured with a head-mounted eye tracker and therefore used in this paper.

To create the dataset, twenty participants (18 men, two women) were asked to track target points on a video display while wearing an Applied Science Laboratories Mobile Eye™ infrared monocular recording device. The participants' right

eye is centered in the video frame and is annotated for specific target points. The dataset is composed of 20 subjects with ages ranging from 21 to 54. The dataset is annotated with head gaze and eye pose information. Eye images are recorded with a resolution of 768 X 480 pixels at 29.97 Hz frame rate.

2) Low resolution datasets for smart devices.

Datasets for low resolution images are more common, for example the MPII-Gaze dataset[29], the UT Multiview dataset[30], and the Gaze Capture[3] dataset are all suitable for training appearance based Deep neural networks. That said, all of these methods come from different distributions and sometimes have incompatible annotations.

For experiments with non-AR/VR augmented spaces where the camera may be far from the user we use the MPII-Gaze dataset[29]. MPII-Gaze was captured over several days and includes annotations for 15 subjects in unconstrained situations in multiple illumination conditions and gaze angles. A Rodrigues transformation is used to map gaze vectors to angles.

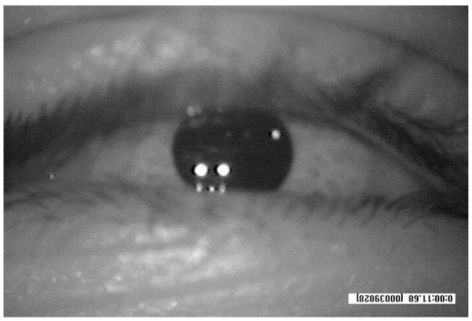


Fig 1. Example image from the McMurrough dataset [28], showing subject's right eye. This is typical of AR/VR systems that utilize eye facing cameras to estimate gaze.

Fig 2. Example image from the MPII-Gaze dataset [29], showing subject's right eye typical of gaze tracking systems that use low quality cameras that are further from the subject.

C. Deep Learning model

Deep learning models such as those proposed by [31] have suggested architectures that work well for low resolution eye gaze estimation systems but these approaches may not be as suitable for high quality eye images as traditional approaches. In this paper, the Deep learning approach is developed for high resolution, close up images of the human eye and compared with one or more traditional approaches.

The inputs to the convolutional neural networks are eye frames from the PoG Eye Tracking Dataset scaled down to 157 X 96 pixels.

Two architectures are developed and tested. The first consist of 5 (3x3) convolutional layers with RELU activation functions, followed by one or more fully connected layers and are trained to perform a regression task, predicting gaze targets given a head pose and eye image as inputs. Next a Resnet 18-like [32] architecture are trained and compared with the previous model.

Lastly, the two above models are compared with a traditional approach, thus providing information about how an appearance-based end-to-end deep learning approach compares with the best alternative.

TABLE II. SUMMARY OF EYE TRACKING METHODS USING LOW RESOLUTION IMAGES

Citation	Input resolution/ eye crop size	Details of method used	Accuracy	Tolerance
[33]	15x40	*ANN based, trained with 2000 images*	*1.7 deg*	---
[10]	640x480 (30 fps video)	*Iris center detection, ellipse fitting, eye corner detection*	1.33 deg	Moderate head movements
[34]	40x30	Image reconstruction using bilinear interpolation, Hough transform	1.89 deg	---
[35]	640x480 (16 fps video)	Fourier descriptors of eye shape, classification with SVM	*90% accuracy*	Head motion lighting variations
[36]	5x5	Images from multiple miniature low resolution eye cameras with ANN	*2.25 deg*	Head motion

D. Deep learning for Eye tracking non-AR/VR applications

Non AR/VR approaches to eye gaze estimation require solutions that can make use of low quality eye images taken at a distance (figure 2) from the user. Traditional gaze estimation methods are unable to perform reliably in this task for unconstrained situations (i.e., outside of lab settings or carefully measured environments). Because traditional methods fail in such situations, approaches based on Deep Learning have recently become popular. These methods use convolutional neural networks, which are can generally be expected to at the level of a human expert on vision tasks when given enough samples.

In augmented spaces, such networks have the additional requirement of real time execution which involves a tradeoff between network complexity and power consumption.

Table II contains examples of networks that have been used for the eye gaze estimation task. It also gives an idea about existing methods that work on low resolution images. Column 2 provides the typical resolution of eye images used for these works. Column 4 mentions the obtained accuracies and Column 5 presents the operating conditions the algorithms can withstand while maintaining sufficient accuracy.

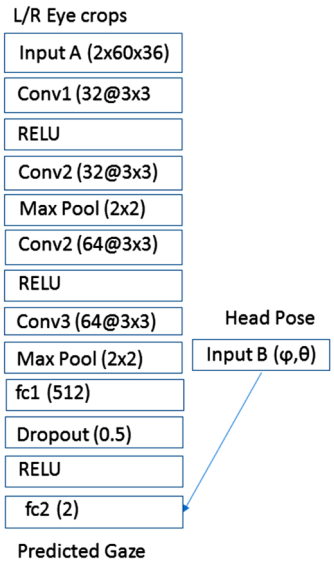


Fig. 3 Architecture of one proposed network from [31].

IV. Experiments & Methodology

A. Data pre-processing

In this paper, the major focus is the presentation of experiments and results from deep learning based gaze estimation using the "near eye" gaze and eye image data provided by the McMurrough et al dataset[28], that is typical of AR/VR devices as described above. The other case of remote gaze estimation using deep learning that uses the MPII Gaze dataset has been discussed in detail in other work [31] and will be presented during the conference.

The McMurrough (or PoG) dataset provides eye images (video frames), gaze target coordinates as well as head pose information for twenty users which are used as inputs in the experiments for this work. In first half of the dataset, the users are asked to keep their head still while in the other, free head movement is allowed. For the first experiments, data of users with limited head movement is used and in the next phase of experiments, data having free user head movements will be used and results will be compared. Unfortunately, this dataset does not provide enough information to utilize pose in the deep learning task, which limits the possible accuracy of the technique.

B. Experiment details

A series of experiments were conducted to determine appearance based convolutional neural network architectures that show the most promise for future exploration. In this section a summary of these experiments is provided. All experiments were trained to perform a regression task, mapping the pixels from the camera facing eye crops to the x and y coordinates that were being gazed upon. Eye crops were reduced to $1/4^{th}$ their original size. All Xception[37] experiments used the Adam optimizer whereas all the other experiments used Stochastic Gradient Descent(SGD). This was because the Xception models failed to converge with SGD while the other networks suffered poor convergence with Adam.

Images from the first 17 people in the dataset were used for training while the remaining were reserved as a test set. To the best of our knowledge this is the first work that utilized the Xception model for AR/VR eye gaze situations.

It may be noted that in this work, all input gaze location values are uniformly scaled between 0 and 1 based on the maximum value of x or y in the labels. Therefore, to get the average distance in pixel space for the outputs x or y, the respective network output values must be multiplied by 1366.0 which is the maximum label value in the dataset. The accuracy in pixels from the method can therefore be stated as:

$$x \text{ (or y pixel deviation)} = \text{output x (or y)} * 1366 \quad (1)$$

As mentioned in the paper describing the PoG dataset[28], the monitor where gaze of participants was tracked has a 32 inch screen and the estimated dot pitch is 0.5mm. Hence the gaze tracking accuracy results from our algorithm using this dataset may be estimated as the deviation from the target location in millimeters (mm) as:

$$x \text{ (or y) deviation in mm} = \text{output x(or y)} * 1366 * 0.5 \quad (2)$$

C. Experiment 1

In this experiment a network utilizing 5x5 kernels and RELU activation functions with 3 convolutional layers, separated by max pooling layers, followed by a fully connected RELU layer and dropout was used. The first convolutional layer had 15 features, the second convolutional layer had 30 features, and the final convolutional layer had 10. This network was trained for 10 epochs.

D. Experiment 2

In this experiment a model utilizing a decreasing number of units in each layer was used. The first 3 layers use 128 3x3 convolution followed by RELU activation functions. These 3 layers are followed by a dropout layer and a max pooling layer. The next block of layers consists of 2 convolutional layers with 64 units, using 3x3 kernels and RELU activation functions. This is again followed by the dropout and max pooling technique before the final block of convolutional layers. This final block of 2 convolutional layers has 32 3x3 kernels with RELU. Finally, a max pooling layer which leads to a fully connected RELU layer with 1024 units before being passed to a final linear layer with 2 outputs (x and y). This network was trained for 10 epochs.

E. Experiment 3

A very small network was developed with a similar architecture to the first experiment. Eight 7x7 convolutional kernels were in the first layer, followed by RELU and max pooling layers. The next convolutional layer had 16 units with 5x5 convolutional kernels, RELU activations and another max pooling layer. The final convolutional layer had eight 5x5 convolutional units with RELU and max pooling. Finally, a fully connected RELU layer and a dropout layer is used before the final linear layer with 2 outputs (x and y). This network was trained for 10 epochs.

F. Xception experiments

This subsection details the experiments performed with the Xception network. For these experiments the last two layers were removed and replaced by a fully connected RELU layer with 1024 units followed by a linear layer of 2 units (x,y). Due to the complexity and size of this architecture it is not reproduced in this paper. Instead we refer readers to the relevant literature[37]. In these experiments the modified Xception network was trained for 10,20,40,60 and 100 epochs with a learning rate of 0.0001 with the Adam Optimizer in Keras[38].

V. Results

The results from the experiments described above are presented in the subsections below. The results are mentioned both as direct outputs from the network as well as in units of centimeter, obtained using the calculations in Section IV. B.

A. Results from experiment 1

Surprisingly, experiment 1, despite being the second smallest network performed competitively with Xception at 10 epochs of training. This experiment resulted in a x axis error of 0.1960 (or 13.3 cm) and an y axis error of 0.1247 (8.5cm).

B. Results from experiment 2

Despite being significantly larger, this network underperformed the one from Experiment 1, with an average x error of 0.2319 (15.8 cm) and average y error of 0.1280 (8.7 cm). Interestingly, while the x error was less than that from Experiment 1, the y error is very similar

C. Results from experiment 3

This experiment resulted in an average error of 0.3181 (21 cm) on x and 0.1808 on the y axis (12.3 cm). This small network helps to establish a lower bound on the number of layers and units for reasonable results indicating that attempting to train networks smaller than this on this dataset is not likely to succeed. Still the similarity of this network to the one in Experiment 1 mean that only a few more neurons are sufficient to greatly increase the accuracy.

D. Results of Xception experiments

The first Xception experiment followed the pattern from the previous experiments with just 10 epochs and a learning rate of 0.0001, resulting in an x error of 0.2800 (19 cm) and a y error of 0.1088 (7.4 cm) . Increasing this to 20 epochs resulted in 0.1962 (13.4 cm) error on x and 0.0977 (6.6 cm) error on y. Running an additional 40 epochs with a reduced learning rate (0.00001) did not show improvement but instead resulted in an x axis error of 0.2238 (15 cm) and an y axis error of 0.1143 (7.8 cm).

The third best results were obtained after 60 epochs at a 0.0001 learning rate. These results were: 0.1356 (9.2 cm) on x and 0.0618 (4.2 cm) on y. The second best results were from using 100 epochs with an average x error of 0.0935 (6.3 cm) and average y error of 0.0466 (3.1 cm).

Finally, the best results were obtained after 1600 epochs. In this experiment the average vertical error was 13.29 mm (1.329cm) and the average horizontal error was 42.2468 mm (4.22468 cm)

VI. Summary & Discussions

It is noticed that in all experiments done with CNN models and the PoG gaze dataset, the error on the y direction is lower than on the x, although we do not currently have an explanation for this. It may result from the difference in the monitor's (where the user gaze was captured during collection of the PoG dataset) horizontal and vertical resolution but this is not sufficient to explain the accuracy differences between horizontal and vertical predations. However, Pose drift is the most likely explanation.

In this dataset, roughly half of participants wear glasses so it remains a matter of investigation as to if this factor has any impact on the result.

One consideration when comparing or evaluating these results is that the average accuracy includes frames where eyes are not open. This is a deliberate choice, as CNNs may be capable of estimating gaze of even closed eyes therefore comparing these results with those that disregard closed or partly open eyes would be misleading. An expanded CNN may utilize temporal information to increase the accuracy of these results as in [39].

The results for CNN based remote eye tracking using data from MPIIGaze dataset are promising and will be presented during the conference.

VII. Conclusion

This paper evaluates the use of deep neural networks for end to end mapping of eye images to gaze coordinates with applications in augmented spaces. The eye images are sourced from two small low cost eye facing cameras: one for the left and one for right eye in the case of AR/VR type systems. For the case of gaze tracking in augmented spaces that do not involve AR/VR, images from one or more cameras at a greater distance are used as input.

As deep learning begins to surpass traditional techniques in many eye gaze tasks, it is of interest to investigate its' potential for gaze estimation on AR/VR setups. Although the error (1.329cm x 4.2246cm) of the best trained model would make it suitable for many gaze estimation tasks, the pure CNN model still underperforms traditional methods.

Unfortunately, accuracy estimates are limited by the fact that this dataset does not provide enough information to utilize pose in the deep learning task and there are currently no suitable alternatives which provide this information.

In future work, problems arising from the lack of suitable datasets for AR/VR systems are addressed by introducing artificial variations into an existing gaze dataset. However, since such augmentations may fail to capture the true distribution of the data, additional real data from an AR/VR setup may be necessary to learn further details about challenges of eye tracking in AR/VR environments. As mentioned in section III.A, there are currently no suitable public datasets that utilize two head-mounted eye-facing cameras of left and right eyes, and therefore a future work may involve creating such a dataset specifically built to

implement deep learning models for augmented spaces. This problem does not exist however, for low quality images taken at a distance which makes training deep learning systems in their use easier.

REFERENCES

[1] D. W. Hansen and Q. Ji, "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 478-500, March 2010.

[2] Naqvi, R.A.; Arsalan, M.; Batchuluun, G.; Yoon, H.S.; Park, K.R. Deep Learning-Based Gaze Detection System for Automobile Drivers Using a NIR Camera Sensor. *Sensors* **2018**, *18*, 456.

[3] K. Krafka *et al*., "Eye Tracking for Everyone," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Las Vegas, NV, 2016, pp. 2176-2184..”

[4] J. Lemley, S. Bazrafkan and P. Corcoran, "Smart Augmentation Learning an Optimal Data Augmentation Strategy," in *IEEE Access*, vol. 5, pp. 5858-5869, 2017..

[5] D. Surie, S. Partonia and H. Lindgren, "Human Sensing Using Computer Vision for Personalized Smart Spaces," *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*, Vietri sul Mere, 2013, pp. 487-494..

[6] A. Shahzada, "A Comprehensive Framework for the Development of Dynamic Smart Spaces," *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Florence, 2015, pp. 927-930.

[7] M. Raeiszadeh and H. Tahayori, "A novel method for detecting and predicting resident's behavior in smart home," *2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*, Kerman, 2018, pp. 71-74.

[8] P. M. Corcoran, F. Nanu, S. Petrescu and P. Bigioi, "Real-time eye gaze tracking for gaming design and consumer electronics systems," in *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 347-355, May 2012..

[9] T. Toyama, A. Dengel, W. Suzuki, and K. Kise, “Wearable reading assist system: Augmented reality document combining document retrieval and eye tracking,” *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, pp. 30–34, 2013.

[10] A. George and A. Routray, "Fast and accurate algorithm for eye localisation for gaze tracking in low-resolution images," in *IET Computer Vision*, vol. 10, no. 7, pp. 660-669, 10 2016.

[11] T. Pfeiffer, “Towards Gaze Interaction in Immersive Virtual Reality : Evaluation of a Monocular Eye Tracking Set-Up,” *Virtuelle und Erweiterte RealitatFunfter Work. der Gifachgr. VRAR*, pp. 81–92, 2008.

[12] J. Y. Lee, H. M. Park, S. H. Lee, T. E. Kim and J. S. Choi, "Design and Implementation of an Augmented Reality System Using Gaze Interaction," *2011 International Conference on Information Science and Applications*, Jeju Island, 2011, pp. 1-8..

[13] S. Bazrafkan, A. Kar, and C. Costache, “Eye Gaze for Consumer Electronics: Controlling and commanding intelligent systems.” *IEEE Consum. Electron. Mag.*, vol. 4, no. 4, pp. 65–71, 2015.

[14] Jason Orlosky, Takumi Toyama, Kiyoshi Kiyokawa, and Daniel Sonntag. 2015. ModulAR: Eye-Controlled Vision Augmentations for Head Mounted Displays. *IEEE Transactions on Visualization and Computer Graphics* 21, 11 (November 2015), 1259-1268.

[15] M Mihai Bâce, Teemu Leppänen, David Gil de Gomez, Argenis Ramirez Gomez:ubiGaze: ubiquitous augmented reality messaging using gaze gestures. SIGGRAPH ASIA Mobile Graphics and Interactive Applications 2016: 11:1-11:5.

[16] P. Renner and T. Pfeiffer, "Attention guiding techniques using peripheral vision and eye tracking for feedback in augmented-reality-based assistance systems," *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, Los Angeles, CA, 2017, pp. 186-194.

[17] T. Piumsomboon, G. Lee, R. W. Lindeman and M. Billinghurst, "Exploring natural eye-gaze-based interaction for immersive virtual reality," *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, Los Angeles, CA, 2017, pp. 36-39.

[18] N. Sidorakis, G. A. Koulieris and K. Mania, "Binocular eye-tracking for the control of a 3D immersive multimedia user interface," *2015 IEEE 1st Workshop on Everyday Virtual Reality (WEVR)*, Arles, 2015, pp. 15-18.

[19] J Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, Matthias Nießner: FaceVR: Real-Time Facial Reenactment and Eye Gaze Control in Virtual Reality. CoRR abs/1610.03151 (2016)

[20] Michael Stengel, Steve Grogorick, Martin Eisemann, Elmar Eisemann, and Marcus A. Magnor. 2015. An Affordable Solution for Binocular Eye Tracking and Calibration in Head-mounted Displays. In *Proceedings of the 23rd ACM international conference on Multimedia* (MM '15). ACM, New York, NY, USA, 15-24.

[21] H. C. Lee, D. T. Luong, C. W. Cho, E. C. Lee, and K. R. Park, “Gaze tracking system at a distance for controlling IPTV,” *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2577–2583, 2010.

[22] S. Plesnick, D. Repice and P. Loughnane, "Eye-controlled wheelchair," *2014 IEEE Canada International Humanitarian Technology Conference - (IHTC)*, Montreal, QC, 2014, pp. 1-4..

[23] Jeffrey S. Shell, Roel Vertegaal, and Alexander W. Skaburskis. 2003. EyePliances: attention-seeking devices that respond to visual attention. In CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03). ACM, New York, NY, USA, 770-771.

[24] H. Khalife, M. A. Okdeh, A. Hage-Diab, A. Haj-Ali and B. Hussein, "Concussion detection using a commercially available eye tracker," *2017 Fourth International Conference on Advances in Biomedical Engineering (ICABME)*, Beirut, 2017, pp. 1-4.

[25] S. Hillaire, A. Lecuyer, R. Cozot and G. Casiez, "Using an Eye-Tracking System to Improve Camera Motions and Depth-of-Field Blur Effects in Virtual Environments," *2008 IEEE Virtual Reality Conference*, Reno, NE, 2008, pp. 47-50.

[26] B. B. Velichkovsky, M. a. Rumyantsev, and M. a. Morozov, “New Solution to the Midas Touch Problem - Identification of Visual Commands Via Extraction of Focal Fixations,” *Procedia Comput. Sci.*, vol. 39, pp. 75–82, 2014.

[27] H. Deng and W. Zhu, "Monocular Free-Head 3D Gaze Tracking with Deep Learning and Geometry Constraints," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, 2017, pp. 3162-3171.

[28] Christopher D. McMurrough, Vangelis Metsis, Jonathan Rich, and Fillia Makedon. 2012. An eye tracking dataset for point of gaze detection. In *Proceedings of the Symposium on Eye Tracking Research and Applications* (ETRA '12), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 305-308.

[29] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, “Appearance-based gaze estimation in the wild,” in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015, pp. 4511–4520..

[30] Y. Sugano, Y. Matsushita, and Y. Sato, “Learning-by-synthesis for appearance-based 3D gaze estimation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1821–1828, 2014.

[31] J. Lemley, A. Kar, A. Drimbarean and P. Corcoran, “Efficient CNN Implementation for Eye-Gaze Estimation on Low-Power/Low-Quality Consumer Imaging Systems,” 2018. arXiv:1806.10890.

[32] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778.

[33] Shumeet Baluja and Dean Pomerleau. 1994. Non-Intrusive Gaze Tracking Using Artificial Neural Networks. Technical Report. Carnegie Mellon Univ., Pittsburgh, PA, USA.

[34] Y. Fu, W. P. Zhu and D. Massicotte, "A gaze tracking scheme with low resolution image," *2013 IEEE 11th International New Circuits and Systems Conference (NEWCAS)*, Paris, 2013, pp. 1-4.

[35] Yu-Tzu Lin, Ruei-Yan Lin, Yu-Chih Lin, and Greg C. Lee. 2013. Real-time eye-gaze estimation using a low-resolution webcam. Multimedia Tools Appl. 65, 3 (August 2013), 543-568.

[36] M. Tonsen, J. Steil, Y. Sugano, and A. Bulling, “Invisibleeye: Mobile eye tracking using multiple low-resolution cameras and learning-based gaze estimation,” Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 1, no. 3, pp. 106:1–106:21, Sep. 2017.

[37] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. arXiv preprint arXiv:1610.02357, 2016.

[38] Francois Chollet ,2015 Keras https://github.com/fchollet/keras. (2015).

[39] Joseph Lemley, Bazrafkan Shabab, Peter Corcoran. “ Transfer Learning of Temporal Information for Driver Action Classification” Proceedings of the 28th Modern Artificial Intelligence and Cognitive Science Conference 2017. March 2017.

Appendix K: Comparison of eye gaze data from different remote eye trackers and experiments

Appendix K

Comparison of eye gaze data from different remote eye trackers and experiments

In this Appendix, sample gaze data from the two different remote eye trackers that were initially tested for gaze data collection as part of this thesis work are presented and compared. The two eye trackers are Tobii EyeX 4C and Eyetribe and their gaze data characteristics for different user distances are observed and discussed in Section K.1. Ultimately, only the Tobii tracker was used based on this comparison, for conducting the full set of experiments done in this thesis.

Using the Tobii eye tracker, gaze data was collected from two user platforms, a desktop and a tablet, for analysis and development of various methods in this thesis work. Sample data collected from the different experiments done on these two platforms are presented in Section K.2 and results from statistical analysis on these data are in Section K.3.

K.1 Comparison of gaze data from two eye trackers

Both the Tobii EyeX 4c and Eye tribe trackers have a specified gaze accuracy of 0.5 degrees and are remote bar-type screen mountable trackers (Figure K.1). For a pilot study with the two trackers, gaze data was collected from the trackers for three different user distances of 45, 60, and 75 cm from a group of 10 participants. The setup included a desktop computer with a monitor (size: 22 inch, screen resolution: 1680 x1050). The trackers were calibrated with their own calibration software (both have 9 point calibration as shown in Figure K.2). Then a common UI (described in Chapter 4) was presented to the participants and their gaze data was collected. The trackers were operated in separate sessions.

Figure K.1 The Tobii EyeX4C tracker (left) and the Eyetribe tracker (right)

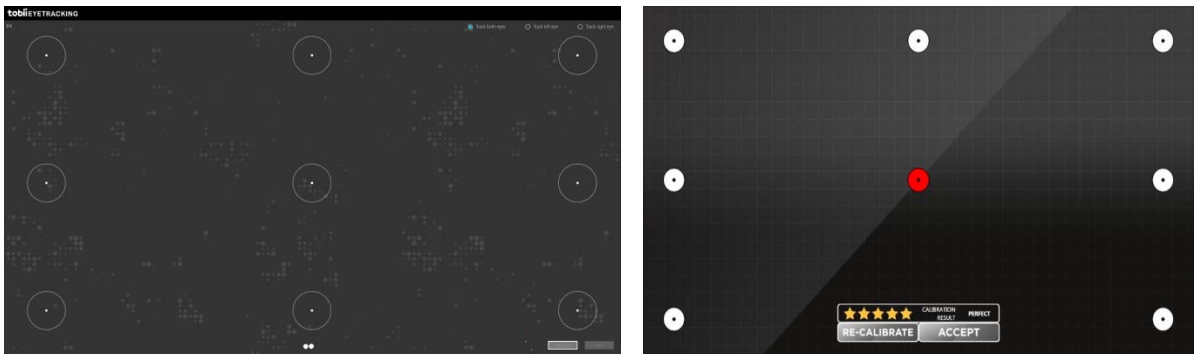


Figure K.2 Calibration screens of the Tobii EyeX4C eye tracker (left) and the Eyetribe eye tracker (right)

Participants were seated at sequentially increasing distances of 45 cm, 60 cm and 75 cm from the tracker-screen setup. Their chin was fixed with a chin-rest and they were asked to gaze at specific stimuli targets that appeared on the display screen as their gaze was recorded by the trackers. The data collection process and stimuli details have been discussed in Chapter 4 of this thesis. The following parameters were calculated for each user for each user-tracker distance.

a. Gaze positions (in mm) data vs ground truth data (locations of stimuli dots)

b. Gaze yaw, pitch angles vs time and corresponding ground truth yaw, pitch angles vs time (ms).

Values of these gaze variables are derived using the equations presented in the paper of Appendix D.

The plots below show the above parameters calculated from raw gaze data collected from the Tobii and Eyetribe eye-trackers. Data from Tobii is on the left and data from Eye tribe is on the right. It was found that for Eyetribe there was no tracking after a user-tracker distance of 60 cm and so gaze data was collected for only two distances 45 and 60 cm for this tracker. The plots below show gaze data variables in black, and ground truth data (stimuli locations and angles) is in blue.

Figure K.3a-j: Plots for gaze data vs ground truth for Tobii and Eyetribe trackers

User1: Tobii EyeX 4C gaze data for 45 cm

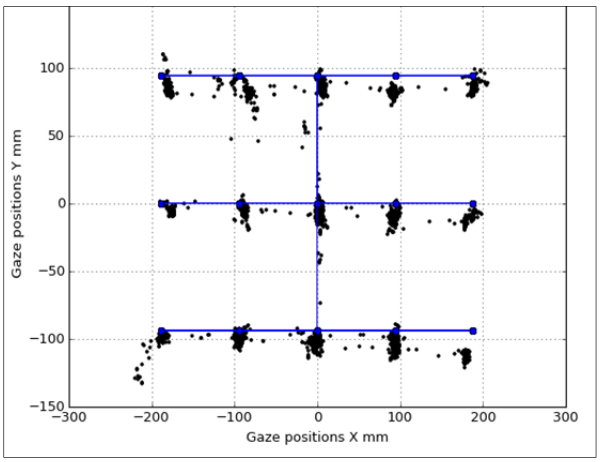


Figure K.3a: Tobii 45cm user1

User1: Eyetribe gaze data for 45 cm

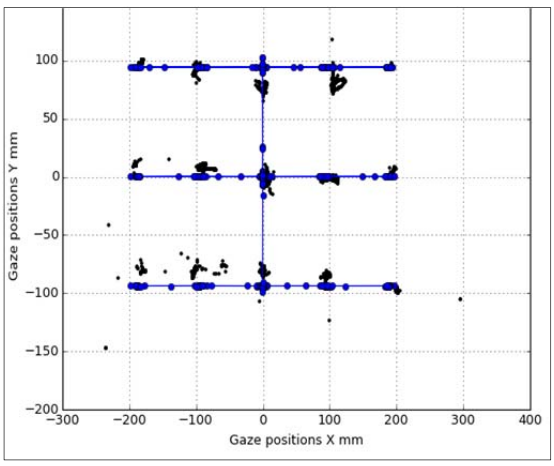


Figure K.3b: Eyetribe 45cm user1

User1: Tobii EyeX 4C gaze data for 60 cm

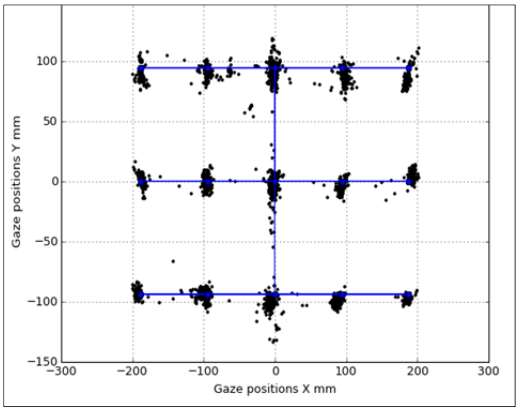


Figure K.3c: Tobii 60cm user1

User1: Eyetribe gaze data for 60 cm

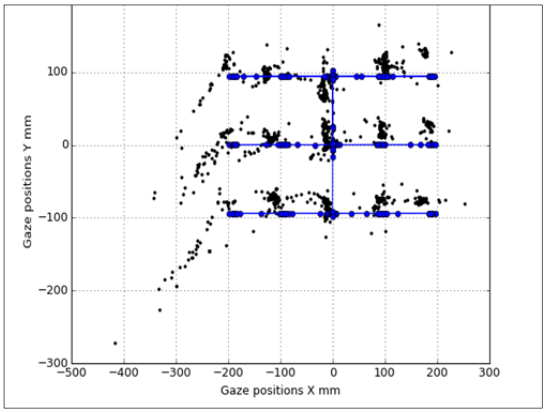


Figure K.3d: Eyetribe 60 cm user1

User1: Tobii EyeX 4C gaze data for 75 cm

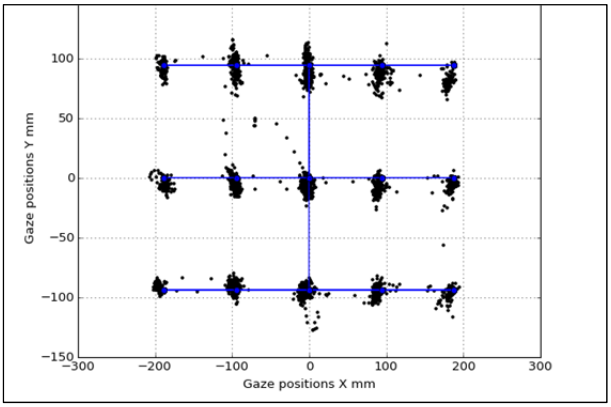


Figure K.3e: Tobii 75 user1

No eye tracking for Eyetribe tracker after 60 cm

User2: Tobii EyeX 4C gaze data for 45 cm

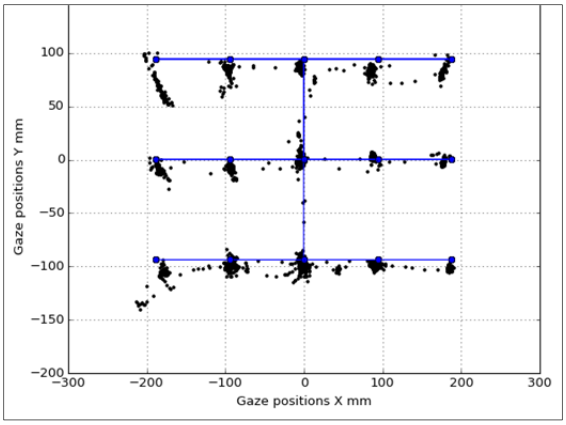


Figure K.3f: Tobii 45cm user2

User2: Eyetribe gaze data for 45 cm

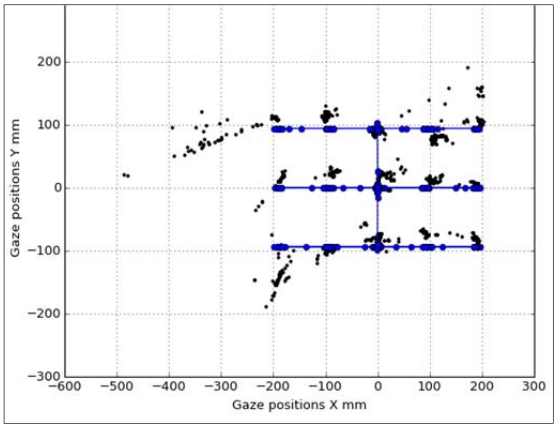


Figure K.3g: Eyetribe 45cm user2

User2: Tobii EyeX 4C gaze data for 60 cm

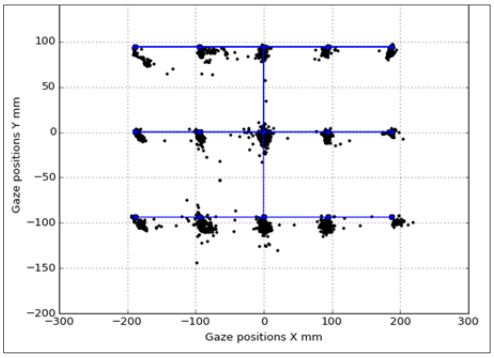


Figure K.3h: Tobii 60 cm user2

User2: Eyetribe gaze data for 60 cm

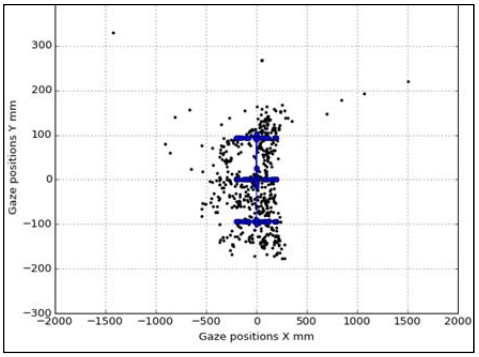


Figure K.3i: Eyetribe 60 cm user2

User2: Tobii EyeX 4C gaze data for 75 cm

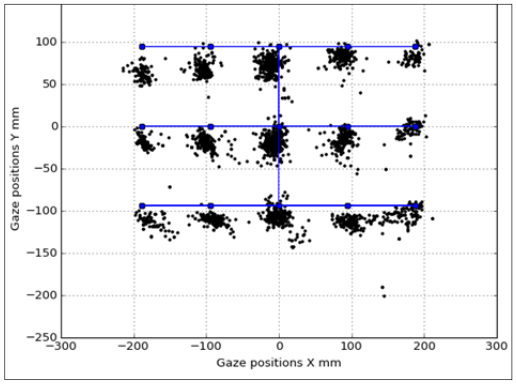


Figure K.3j: Tobii 75cm user 2

No eye tracking for Eyetribe tracker after 60 cm

Figure K.4a-d: Gaze vs ground truth yaw angles for Tobii and Eyetribe trackers

Tobii: Gaze (blue) and ground truth (black) yaw angles

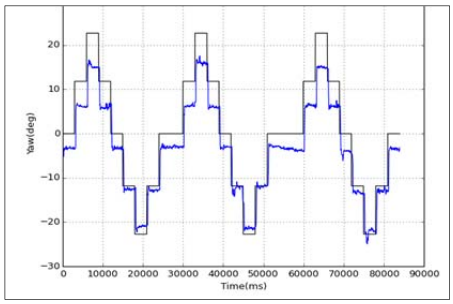


Figure K.4a: 45cm

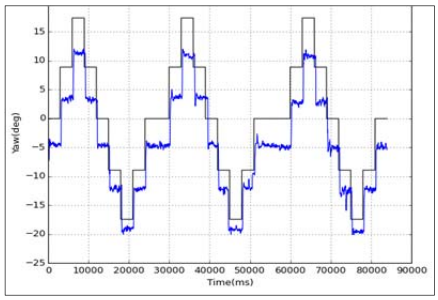


Figure K.4b, 60cm

Eyetribe: Gaze (blue) and ground truth (black) yaw angles

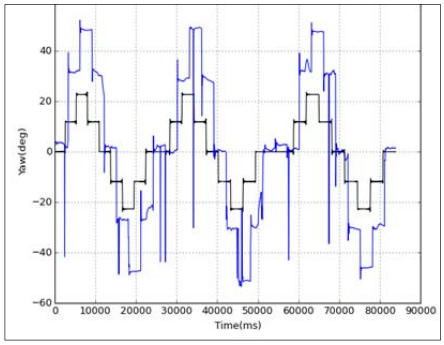


Figure K.4c: 45cm

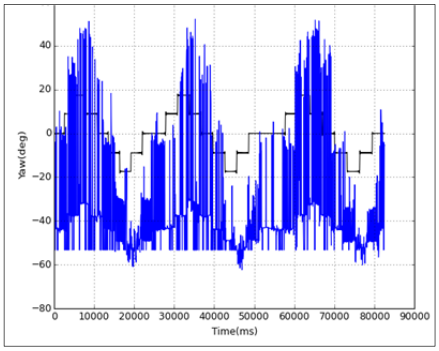


Figure K.4d: 60cm

Figure K.5a-d: Plots for gaze vs ground truth pitch angles for Tobii and Eyetribe trackers

Tobii: Gaze (blue) and ground truth (black) pitch angles

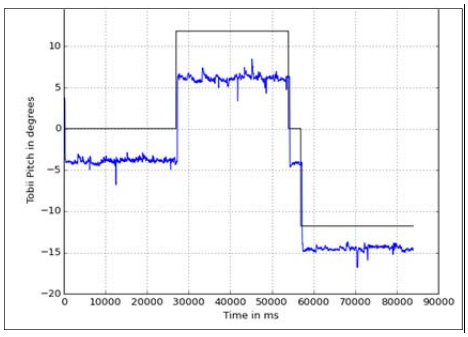


Figure K.5a: 45cm

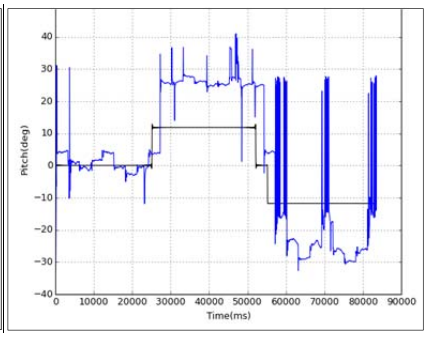


Figure K.5b: 60cm

Eyetribe: Gaze (blue) and ground truth (black) pitch angles

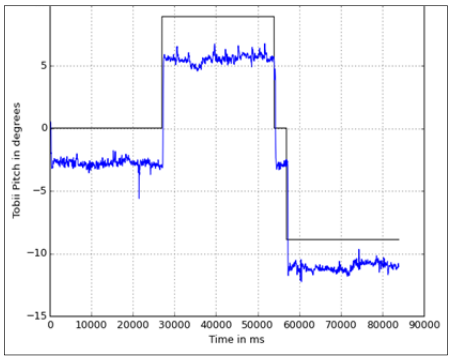


Figure K.5c: 45cm)

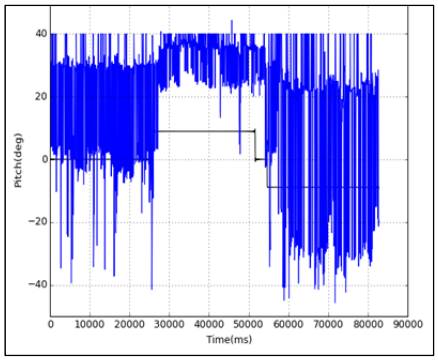


Figure K.5d: 60cm

Discussions

It was observed that gaze data from the Eyetribe tracker was generally noisy for all the 10 participants (data from 2 of whom are included above). Also above a user distance of 60 cm there was no tracking for Eyetribe whereas the Tobii tracker was seen to produce consistent results till 85 cm. Also, the Eyetribe tracker lost device connection several times during data collection sessions and had to be restarted. For these reasons, the Tobii eye tracker was selected for the full set of eye tracking experiments done as part of this thesis, which are described in Chapter 4.

K.2 Sample data from different eye tracking experiments

As discussed in Chapter 4, Section 4.1 of this thesis, a series of gaze data collection experiments were done as part of this thesis work on a desktop and tablet platform using the Tobii eye tracker. These experiments included a) user distance experiments where users were seated at 50, 60, 70, 80 cm from the tracker. This was done for both the desktop and the tablet platforms b) head pose experiments where a user had to position their head at certain fixed head pose angles while their gaze data was collected. This was done only for the desktop platform c) platform pose experiments where the eye tracking platform or tablet was oriented at certain fixed tablet pose angles while user gaze data was collected. This was done only for the tablet platform. Below, sample data from these experiments are

provided and then results from statistical analysis of data from different experiments are presented. In the figures, gaze data is shown as the black points while the ground truth data are the blue lines.

Figure K.6a-d: Data from user distance experiments: Desktop

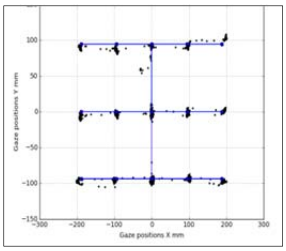


Figure K.6a: 50 cm

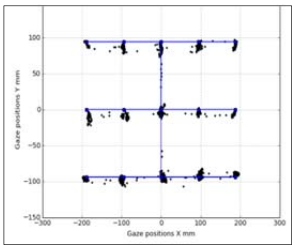


Figure K.6b: 60 cm

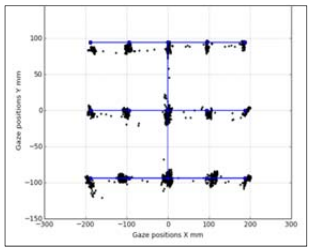


Figure K.6c: 70 cm

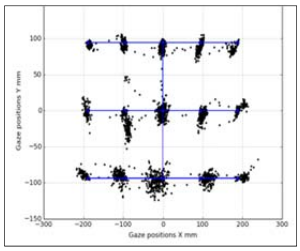


Figure K.6d: 80 cm

Figure K.7a-q: Data from head pose experiments: Desktop

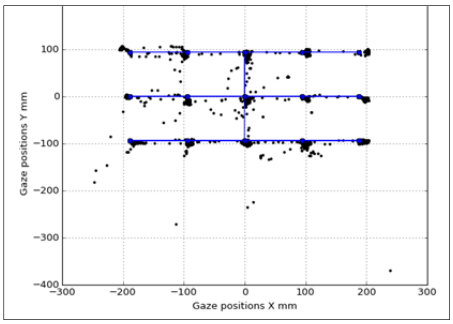


Figure K.7a : Head pose neutral

Head roll plus

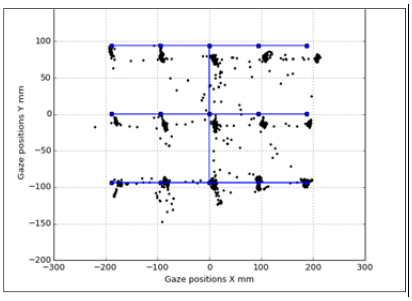


Figure K.7b: Plus 10 degree

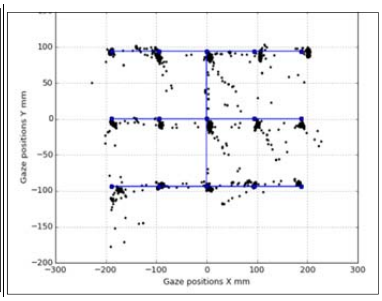


Figure K.7c: Plus 20 degree

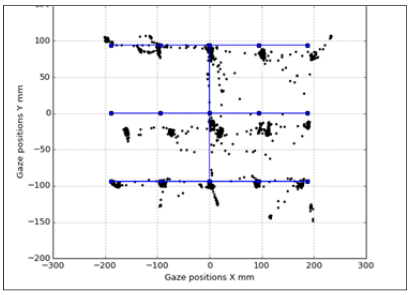


Figure K.7d: Plus 30 degree

Head roll minus

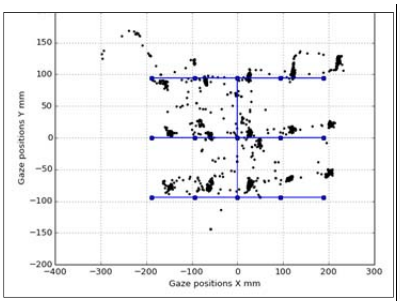


Figure K.7e: Minus 10 degree

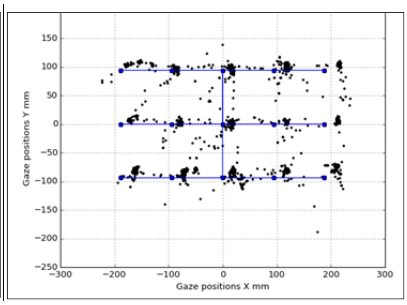


Figure K.7f: Minus 20 degree

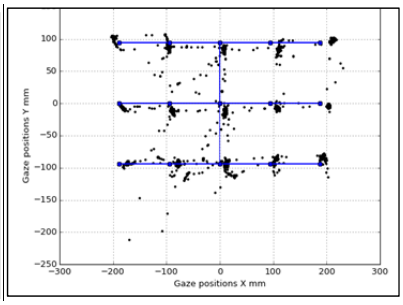


Figure K.7g: Minus 30 degree

Head yaw plus

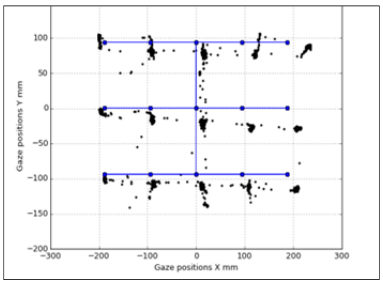


Figure K.7h: Plus 10 degree

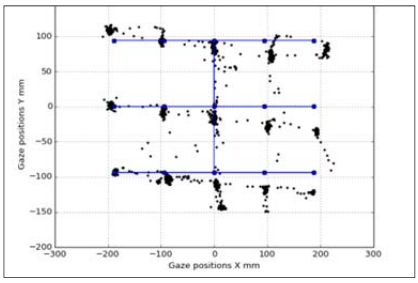


Figure K.7i: Plus 20 degree

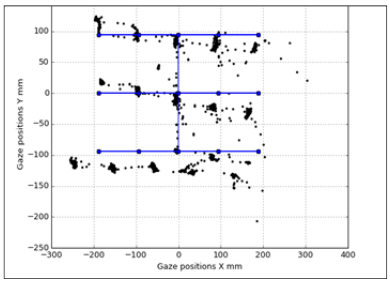


Figure K.7j: Plus 30 degree

Head yaw minus

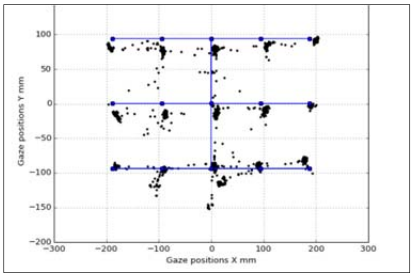


Figure K.7k: Minus 10 degree

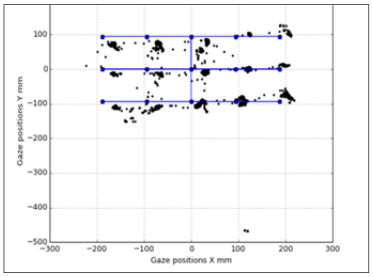


Figure K.7l: Minus 20 degree

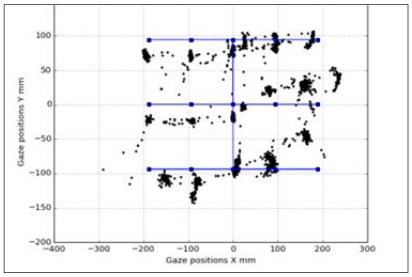


Figure: K.7m: Minus 30 degree

Head pitch plus

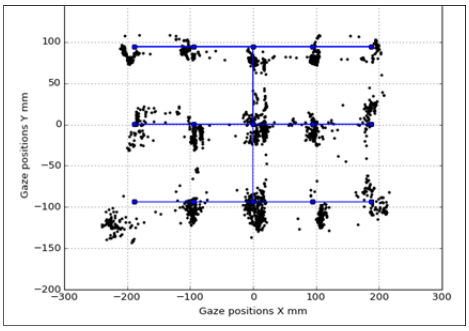


Figure K.7n: Head pitch plus 10 degree

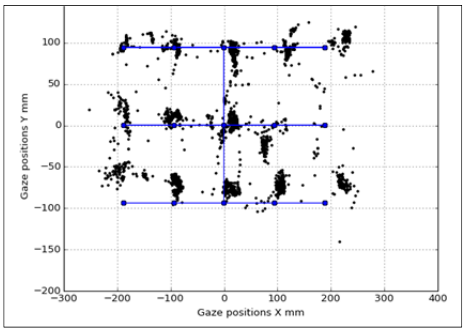


Figure K.7o: Head pitch plus 20 degree

Head pitch minus

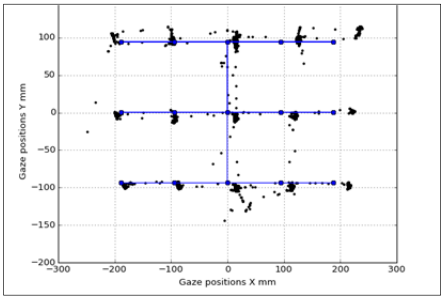


Figure K.7p: Head pitch minus 10 degree

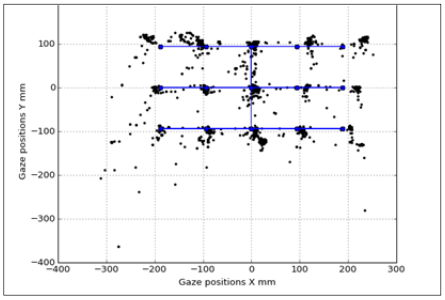


Figure K.7q: Head pitch minus 20 degree

Figure K.8a-d: Data from user distance experiments: Tablet

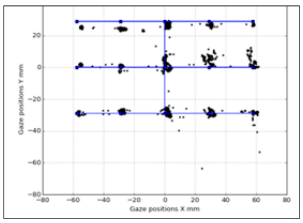


Figure K.8a: 50 cm

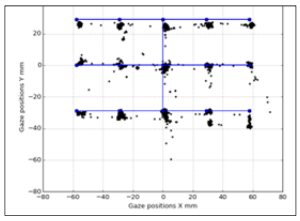


Figure K.8b: 60 cm

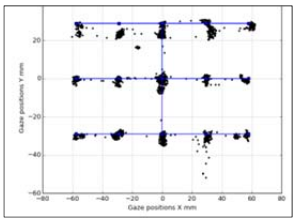


Figure K.8c: 70 cm

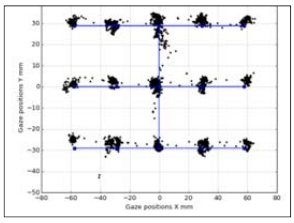


Figure K.8d: 80 cm

Figure K.9a-g: Data from tablet pose experiments

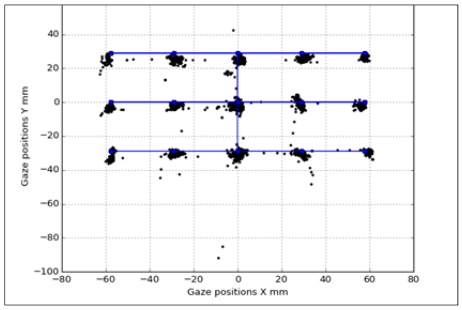


Figure K.9a: Tablet pose Neutral

Tablet roll plus/minus

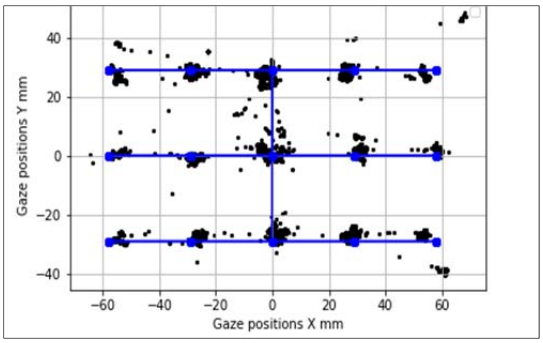


Figure K.9b: Tablet roll plus 20 degrees

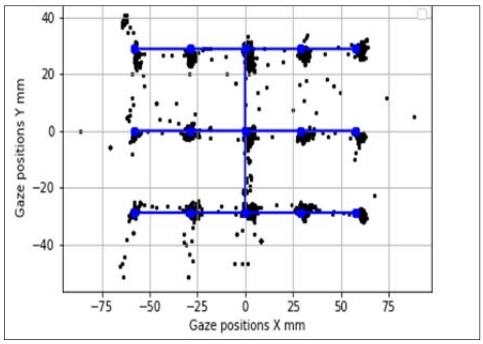


Figure K.9c: Tablet roll minus 20 degrees

Tablet yaw plus/minus

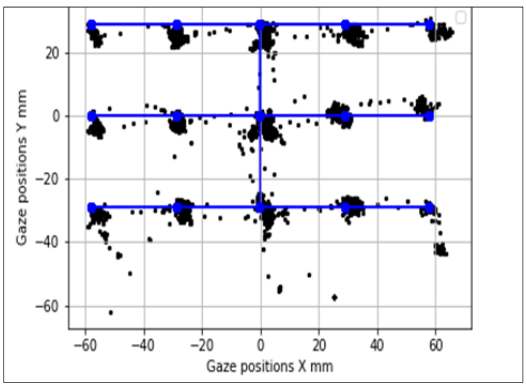


Figure K.9d: Tablet yaw plus 20 degrees

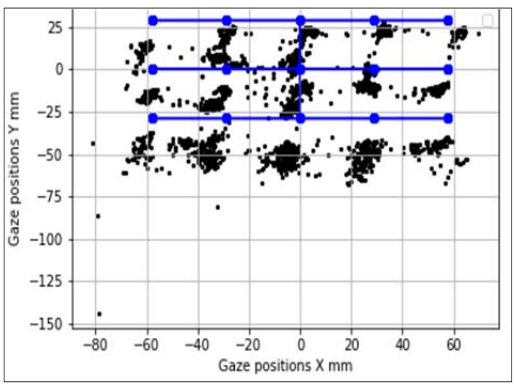


Figure K.9e: Tablet yaw minus 20 degrees

Tablet pitch plus/minus

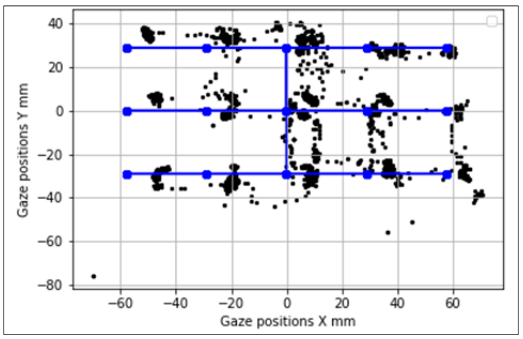


Figure K.9f: Tablet pitch plus 20 degrees

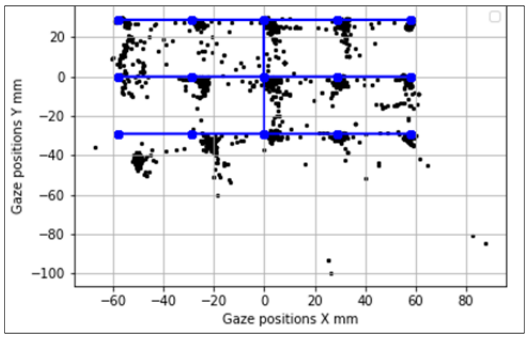


Figure K.9g: Tablet pitch minus 20 degrees

Discussions

In the above figures, where raw gaze data from the different experiments are overlaid on ground truth, it can be seen that data from different operating conditions often look similar. Also no distinct patterns are observable in the data corresponding to different operating conditions. This is partly because of the existence of outliers in the data. In Section K.3, it will be shown that gaze data obtained under different operating conditions do have different error distributions and statistical properties.

K.3 Comparison of gaze data characteristics from different experiments

The tables K.1 and K.2 below presents the gaze error statistical values (after outlier removal) from desktop and tablet experiments respectively. The methods for calculating gaze errors and statistical metrics on gaze error values is provided in the paper of Appendix D and the process of outlier removal and statistical analysis of gaze errors is presented in the paper of Appendix F (Section II.C).

Table K.1. Gaze error statistics from desktop experiments

	UD50	UD60	UD70	UD80	Roll 20	Yaw 20	Pitch 20
Mean	3.37	2.04	1.21	1.02	3.7	8.51	3.15
MAD	3.49	1.77	0.82	0.66	3.63	10.0	1.90
IQR	1.13	0.77	0.76	0.79	1.21	1.49	1.59
95% interval	3.15-3.59	1.90-2.18	1.15-1.26	1.16-1.24	3.30-4.09	7.60-9.43	2.83-3.47

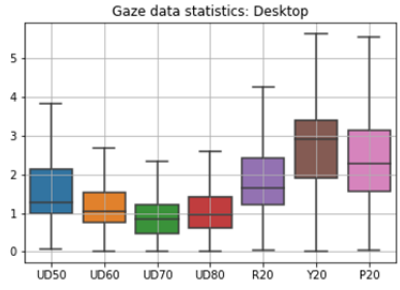


Figure K.10a: Error statistics: Desktop

Table K.2. Gaze error statistics from tablet experiments

	UD50	UD60	UD70	UD80	Roll 30	Yaw 30	Pitch 20
Mean	2.68	2.46	0.59	1.55	7.74	4.25	2.45
MAD	0.38	0.42	0.29	0.24	0.77	0.60	0.46
IQR	0.39	0.54	0.33	0.22	0.75	0.53	0.23
95% interval	2.65-2.71	2.43-2.48	0.57 -0.61	1.53-1.57	7.69-7.80	4.22-4.29	2.41-2.49

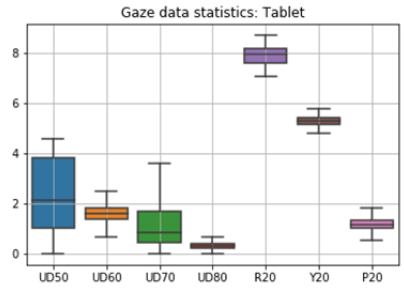
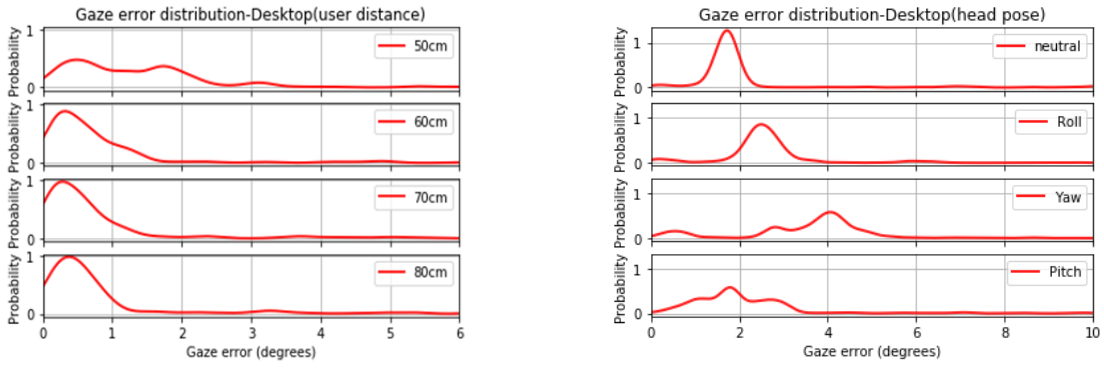


Figure K.10b: Error statistics: Tablet

In Table K.1 and Figure K.10a, UD 50, UD60, UD70, UD80 correspond to gaze data from different user-distance experiments done on the desktop platform and R20, Y20 and P20 correspond to gaze data from head pose roll pitch yaw angle (20 degrees for each) experiments. It is seen that gaze error levels are higher at low user distances and error reduces as user-tracker distance increases. Errors due to head yaw are seen to have the highest magnitude and errors due to head pitch have the highest inter-quartile range (or variability) in error values. Also error levels due to various head poses are quite higher compared to when head pose is neutral (UD60 values in Table K.1). All values in the table have units in degrees of angular resolution.

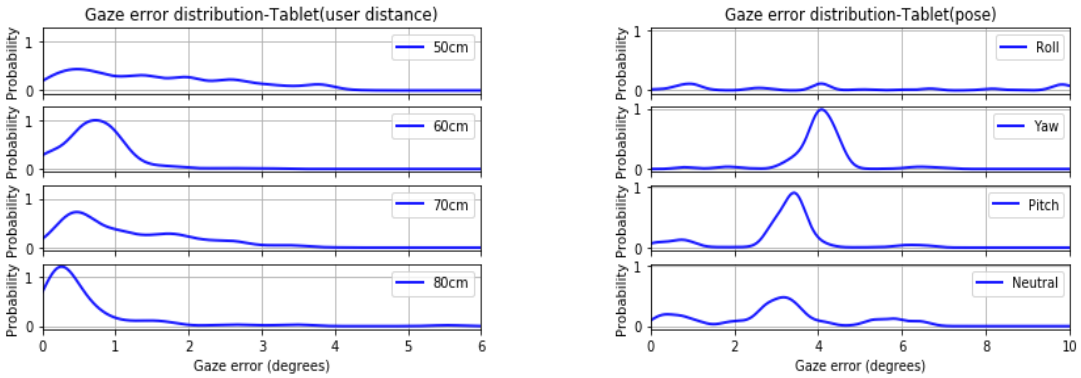
In Table K.2 and Figure K.10b, UD 50, UD60, UD70, UD80 correspond to gaze data from different user-distance experiments done on the tablet platform and R20, Y20 and P20 represent data from the tablet pose roll pitch yaw angles (20 degrees for each) experiments. It is seen that magnitudes of errors due to tablet pose changes are high and the highest error is caused due to platform roll variations. It is also seen that the error characteristics from tablet data are quite different than those from the desktop platform. Compared to desktop data, the error magnitudes are lower for tablet for all user distances. Also magnitudes of errors due to different platform poses (Fig K.10 b) are higher than errors due to head poses (Fig K.10a).

Figure K.11a and K.11b below show gaze error distributions for the data (after outlier removal) from desktop user distance and head pose experiments. The method for estimating gaze error distributions are discussed in the paper of Appendix F. It is seen that each operating condition (e.g. user distance or head pose) leaves a definite signature on the gaze error distributions. Clear distinction exists between patterns of gaze errors for different user distances and head poses as the error distribution shifts towards higher, average or lower error values for different conditions. Similar observations are made for tablet data for different conditions (Figures K.11 c and K.11d). These error patterns are not observable from raw gaze data from the experiments as presented above. Also the error distributions caused due to the impact of different operating conditions are non-Gaussian and do not resemble any known statistical distribution function. These aspects are discussed in the paper of Appendix F.



(a) (b)

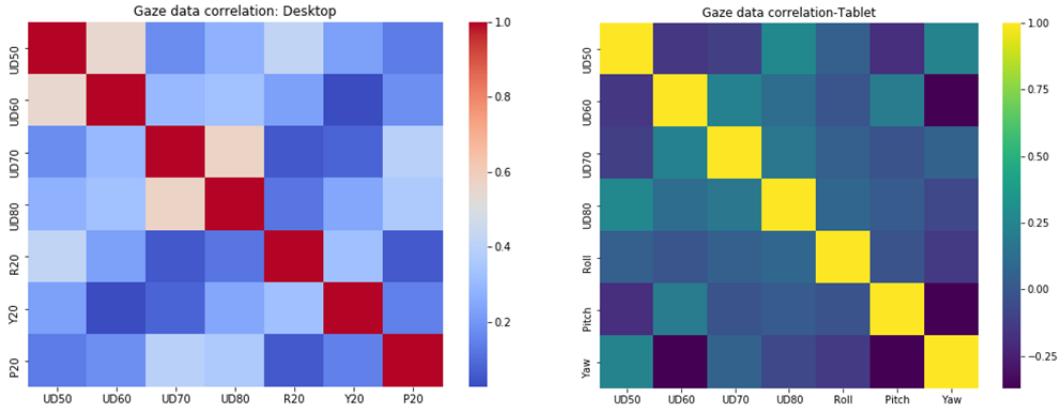
Figure K.11 Gaze error distribution due to: a) user distance –desktop (b)due to head pose- desktop



(c) (d)

Figure K.11 Gaze error distribution due to (c) user distance –tablet (d)due to platform pose- tablet

Correlation studies were also done to observe similarity between gaze errors obtained from different experiments mentioned above. This is presented in Section 2.5.4 of the paper of Appendix F. It is seen from the correlation plots of Figures K.12 a (for data from desktop) and K.12b(for data from tablet) that gaze data collected under different operating conditions from the same platform and eye tracker do not have any correlations between their characteristics.



(a) (b)

Figure.K.12: Correlation between data collected from a) desktop (R20, Y20, P20 refers to head pose) (b) tablet experiments (R20, Y20, P20 refers to tablet pose)