The Report Committee for Nitin Verma
Certifies that this is the approved version of the following report:

Conducting Eye Tracking Studies in Interactive Information Retrieval

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

Jacek Gwizdka

Eric Nordquist

Conducting Eye Tracking Studies in Interactive Information Retrieval

by

Nitin Verma, B.S.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Information Studies

The University of Texas at Austin

May 2017

Acknowledgements

I acknowledge the help I received from my supervisor for this report, Dr. Jacek Gwizdka for offering me the opportunity to work on this project, and for supporting me when I found myself slipping on some deadlines. I would also like to thank Prof. Eric Nordquist for being an inspiring teacher and for sharing his unmatchable knowledge of the psychological, practical, and business aspects of usability studies. Finally I would like to thank Mr. Ilia Korjoukov from Okazolab Ltd for offering continuous help to me in trying to better understand and work with Okazolab's EventIDE stimulus presentation software.

Abstract

Conducting Eye Tracking Studies in Interactive Information Retrieval

Nitin Verma, M.S. Info. Stds.

The University of Texas at Austin, 2017

Supervisor: Jacek Gwizdka

This report seeks to contextualize the field of Interactive Information Retrieval (IIR) in the broader context of Information Retrieval (IR), and the psychological aspects of the human visual system. In this report, I present to the readers a concise background of the fields of IIR, and IR, and experiment design in eye tracking research. After the background and literature review on IIR, I provide the readers a brief tour of the concepts behind eye tracking technology. Finally, I describe the stimulus presentation and control software called EventIDE, which offers numerous capabilities for conducting behavioral research. In particular, I focus on eye tracking, and continuing with the theme of deploying eye tracking to conduct IIR research take the readers through a sample experiment in EventIDE. At the end I analyze specific outcomes of the sample experiment to demonstrate how eye tracking can be used to collect data about users' interaction with the layout and presentation of information on modern digital interfaces.

Table of Contents

List of Figures

Chapter 1: Introduction

This report seeks to explicate the use of Eye Tracking in Interactive Information Retrieval (IIR) research. While the focus of this report is IIR, it is worth taking a look at the field of Information Retrieval (IR) in order to understand the broader context within which IIR research is situated. I begin this report with an exploration of IR that forms a basis from which the fundamental need for eye tracking ultimately arises. Subsequently, I explain what the field of Interactive Information Retrieval (IIR) entails, and how it compares to IR. Thereafter I offer the readers a detailed background and literature review that seeks to contextualize both IR and IIR in an information science context. Building on, I explain the characteristics of human vision, and the workings of the human brain pertaining to visual stimuli. After providing this firm contextual and theoretical background, I move on to practical aspects of conducting IIR research focusing on eye tracking as the key data collection component. These practical aspects include a brief theoretical background on the kinds of research that can be conducted using eye tracking, and general concepts related to experiment design in the IIR domain. Finally, I reinforce the understanding of the theoretical concepts by describing in detail an IIR experiment design and the stimulus presentation and control software program EventIDE. This description of EventIDE includes a walkthrough of the various features of the program, its advantageous features when compared to similar programs, experiment design and setup in EventIDE, and finally the collection and analysis of data output by it.

Chapter 2: What is Information Retrieval (IR)?

Information Retrieval (IR) is a field of research that pertains to understanding and improving how efficacious information systems are in providing the information being sought. While agents that seek information from these systems are most often human users, it is typical that systems also seek specific (or generic) information from one another. Examples of the former, *i.e.*, human agents seeking information, include users seeking help finding the desired resources in a library, or users searching for information on a computer – using web search engines like Google, or the search function on their desktop or smartphone to find files and documents. Examples of systems seeking information from other systems are a web browser requesting dynamically generated HTML documents from a web-server, or a price-comparison website collecting information from online shopping websites. While information-seeking agents are an important part of the overall picture, the focus in most IR research is the core system itself, *i.e.*, the system that interprets agent queries and retrieves information.

The aim of IR research is to build systems that understand the requesting agent's needs and retrieve information that is relevant to those needs. Relevance is a focus area that has received increased amount of attention in the development of digital or web-based IR systems. While relevance is an important *dependent* variable in much of IR research, there are a few *independent* variables that directly or indirectly influence relevance. Some of these influencing variables include classification (or tagging, or categorization) of information objects, correctness and completeness of metadata, indexing and updating of records, and establishment of relevant relationships between different objects or classes of objects. Much research effort has also been spent quantifying or measuring the influence of these variables, as well as the reliability of these measures (Harman, 2011). These

influencing variables provide the basis for what are called ‘relevance criteria’ in IR research. Criteria such as *Topicality, Scope, Tangibility, Recency, and Format* have been identified as some of the most salient criteria for adjudging relevance on the web (Balatsoukas & Ruthven, 2010). The evaluation of relevance in IR systems is done to measure the efficacy & usefulness of these systems, and also to enable testing to better understand the workings of these systems. A firm understanding is fundamental to improving general performance and for making targeted improvements in IR systems (Harman, 2011).

Chapter 3: What is Interactive Information Retrieval (IIR)?

In step with the evolution of the web as the dominant source of information for users, and the increasing availability of digital (and digital-only) content, human users have quite naturally become a focus of research in IR. Since the relevance of information retrieved by most web-pages is ultimately contingent on human judgment, there is a pressing need to better understand relevance from the human brain's perspective. While it is vital to measure relevance (and efficacy) as perceived by the brain, at the same time, it is very difficult to do so directly. This is where systems-centric IR research joins forces with "human focused studies" (Kelly, 2009) to gain insight into how users *interact* with IR systems – hence the term *Interactive* Information Retrieval. Kelly (2009) describes IIR as "IR with users", and places it in the middle of a range extending from system focused studies to human focused studies. Whereas IR studies are more concerned with developing and evaluating system-level algorithm design and indexing techniques (Kelly, 2009), IIR focuses more on users' interactions with IR systems. In other words, whereas retrieval algorithms may be the focal point of most IR research, IIR studies are targeted at *evaluating* how the retrieved information is *presented* to users, and how users interact with that information. This interaction includes, among other things, how users navigate search result pages, and how they refine their queries in case they are not satisfied with the existing set of results. This understanding is crystallized in Kelly's assertion that the way users interact with the system impacts the system's responses (Kelly, 2009), hinting at how some modern IIR systems *suggest* refined queries, or rank results in a particular order.

A good way to begin understanding interactive information retrieval is to break down such interactions conceptually into constituent parts. At a fundamental level, these parts are:

(i) the information need,

(ii) the system to be used for the search,

(iii) the resource(s) to be searched,

(iv) why a resource is chosen,

(v) how the information need is conveyed to the IR system,

(vi) the cognitive activity in the brain of the human information seeker while interacting with the system, and

(vii) the level of satisfaction with the retrieved results.

The last component, level of satisfaction, can further be divided into two parts: the relevance of the results, and the ways in which the searcher is allowed to interact with those results. Ruthven (2008) elaborates the idea of Cool *et al.* (Cool, Park, Belkin, Koenemann, & Ng, 1996) that how the decisions involved in choosing the right resource or system are "complicated because skills learned using one type of system do not always transfer simply to searching a different type of system". This complication is especially relevant in IIR research, because people bring information seeking approaches they have learnt on one system onto other systems (Ruthven, 2008). These previously acquired approaches are analogous to the concept of 'mental models' often invoked in usability research. In usability, as well as in psychology research, it is a well-recognized problem that people form mental models of interactive systems and *try* to apply them to other systems. Many times, these models do not transfer well onto unfamiliar or new interactive systems. Cool *et al.* reason that it is for this difficulty in transferring existing information seeking approaches that researchers face the challenge of designing IR systems that support users' existing strategies and habits (Cool *et al.*, 1996).

As far as IR systems are concerned, their two broad types are query-based and browsing-based systems (Ruthven, 2008). Query-based systems require users to express their information need explicitly, in the form of a query or request; browsing-based systems, on the other hand, “help searchers understand and navigate an information space” (Ruthven, 2008). The interaction between the user and the system is usually mediated by a visual interface. Designing or improving an interface, though, also tends to impact the IR system itself (Ruthven, 2008).

Chapter 4: IIR and Relevance Feedback

An important aspect of IIR research is gathering relevance feedback from users. Relevance feedback can be classified into two broad categories: explicit, and implicit feedback (Kauppi *et al.*, 2015). Writing in terms of relevance *ratings*, Claypool *et al.* define explicit feedback as asking users to tell the system about the information it presented to them (Claypool, Brown, Le, & Waseda, 2001), mentioning that explicit feedback gathering "interrupts normal browsing and reading patterns" (Claypool *et al.*, 2001). Another disadvantage of gathering explicit feedback is that users may lose motivation to provide feedback if they stop perceiving the benefits of providing it (Claypool *et al.*, 2001). In contrast, implicit feedback is much less intrusive, and the techniques used to gather feedback infer relevance by monitoring user behavior (Claypool *et al.*, 2001; Kauppi *et al.*, 2015). Implicit feedback is most often inferred through physiological signals like gaze, heart-rate, skin temperature, and galvanic skin response (mentioned earlier) (Kauppi *et al.*, 2015). In a loose sense, explicit feedback gathering can be compared to interview or questionnaire based research methods, and implicit feedback gathering can be compared to ethnographic methods.

When users are brought in the loop of IR research, the measurement and evaluation of their interactions with IR systems poses a significant challenge. This challenge exists primarily because the cognitive activity in the users' brains is difficult to measure directly. Therefore, researchers have to rely on study techniques such as user interviews, 'think aloud' protocols, and behavioral measurement techniques like mouse movements and fixations, biological measurement techniques like Galvanic Skin Response (GSR), Electroencephalography (EEG), and Eye Tracking. The last technique in that list, Eye Tracking, forms the central theme of this report.

In the remainder of this report I will cover the basics of eye tracking, including its psychological basis, the peculiarities of human vision, and most importantly how to design eye tracking studies specifically for studying scrollable web pages. Also, I will stress on the presentation of information via scrollable web-pages as a subject of eye tracking analysis.

Chapter 5: Background on IIR, and the Human Visual System

Eye tracking, in particular, is an effective means for studying the above mentioned physiological signals. This is in part because of the relatively unobtrusive nature of most eye tracking apparatus, and the fact that the human visual system is the primary site for the brain to collect data for visual stimuli (information presented on electronic displays in our case). The analysis of eye tracking data opens a window into the relevance processing occurring inside the human brain. In simple terms, eye tracking is a research tool that helps us understand visual attention: where, and for how long, do users look at, and the paths that their eyes follow (Schall & Romano Bergstrom, 2014). Compared to "think aloud" protocols, also referred to as *Concurrent Verbal Protocols (CVP)* (Bojko, 2013), in usability research, eye tracking helps researchers infer even those aspects of user experience that users cannot describe verbally (Schall & Romano Bergstrom, 2014). A *Retrospective Verbal Protocol (RVP)*, on the other hand, refers to studies where participants verbalize or describe their experience after the task has been completed (Bojko, 2013). One of the main disadvantages of the CVP is that it increases the cognitive load on the brain, and therefore may interfere with the task itself. It is for this reason, that the RVP, combined with eye tracking may be a better research strategy in user experience research than combining CVP with eye tracking (Bojko, 2013).

One of the most counter-intuitive, and thus remarkable, things about human vision is that even when we claim that our eyes are stable and fixated, eye tracking studies have revealed that our eyes are "constantly moving around to construct a complete picture of what we are looking at" (Schall & Romano Bergstrom, 2014). This movement is essentially because of the small area of the visual field that the human eye can focus with sufficient detail. It is worthwhile to segue into a cursory description of the human visual system.

THE HUMAN VISUAL SYSTEM

For the purposes of this article, I will focus primarily on the human eye while giving due consideration to how the brain interprets the transduced electrical signals from the eyes.

The Structure of the Human Eye

As shown in Figure 1, the human eye consists of the following major components: the cornea, the iris, the pupil, the lens, and the retina. Light incident on the cornea gets focused onto the retina by the lens, and the amount of light that reaches the lens is controlled by the diameter of the pupil.

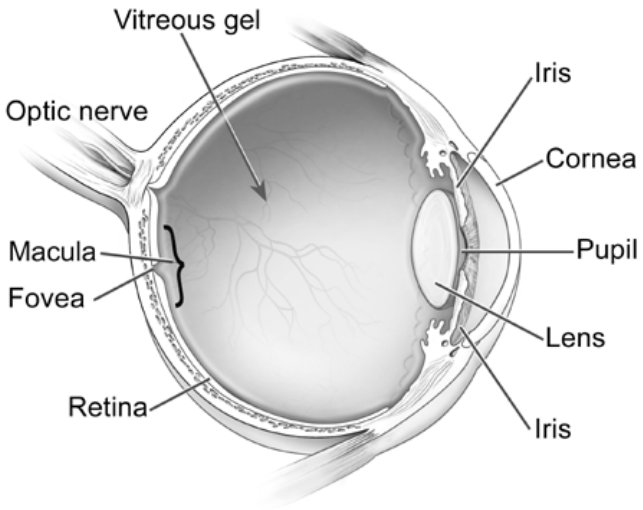

Figure 1: The Structure of the Human Eye (National Eye Institute (NEI), n.d.)

From a psychological perspective, one of the two most important components of the human eye is the pupil. The diameter of the pupil is known to be a reasonably direct

indicator of the cognitive effort being applied by the brain at any given instant – a larger diameter corresponds to situations where the brain is expending more attentional resources, and a smaller diameter corresponds to relatively less attention. Optically, the pupil is what lets the required amount of light enter into the eye, being focused by the lens just behind it. The lensed image is then focused on to the retina which is lined with cells called *rods* and *cones.* These cells convert the incident light, and therefore the whole image, into electrical signals for the brain. The image focused on the retina gets transduced into electrical signals that are carried over to the brain by the optical nerve.

The cones are much more sensitive to spatial frequency (*i.e.*, the number of repetitions per unit distance, rather than time in case of temporal frequency, or simply frequency as measured in Hz) than rods, and are therefore, the primary providers of visual *detail* to the brain (Holmqvist & Nyström, 2011). The rods, on the other hand, are more sensitive to differing levels of brightness or intensity, and contribute to our visual system's ability to detect movement (Biedert, Buscher, & Dengel, 2010). The sensitivity of the cone cells to spatial frequency components is what grants humans color vision.

Structurally, and more importantly for our understanding, cones are highly concentrated in an area of the retina called the *fovea*, and sparsely distributed in the periphery. Therefore, out of the total visual field captured on the retina, only the portion captured by the fovea has the highest sensitivity towards detail. The size of this foveal visual field is approximately equivalent to the size of a thumbnail viewed from an arm's length (Holmqvist & Nyström, 2011), or close to 2° of the visual field (Holmqvist & Nyström, 2011). The fovea is the portion of retina where the image of the object being directly looked at is focused (Bojko, 2013). It suffices to say that at any given moment the eye, and therefore the brain, can only see a very small region of the visual field in full

detail. The visual area just outside of the foveal vision is referred to as *parafoveal* vision, and the area further out is known as *peripheral* vision. The parafoveal region is blurry, and the peripheral region is blurrier still (Bojko, 2013). In specific terms, the overall visual field captured by human eyes is elliptical in shape, and spans approximately 200° in width, and about 130° in height (Biedert *et al.*, 2010).

Eye Movements as Windows to the Brain's Cognitive Activity

The peculiar distribution of cones and rods in the retina, and therefore a limited area over which detail can be captured with full acuity, necessitate movements of the eyes to gather detail about the larger visual field. The small angular width of the foveal region, in combination with eye movements creates a system that scans the scene and 'filters' visual information focusing on the most important features of the visual field. This filtering of information is a very important mechanism, because it saves the brain from being overwhelmed with information (Bojko, 2013). The fact that a significant amount of the brain's 'bandwidth' for vision is used in controlling the muscles that guide eye movement, may also be responsible for the evolution of this movement-based scanning/filtering strategy.

The eye movements involved in scanning the visual field are called *saccades*, and are characterized by the speed at which they are made – "the fastest movements produced by an external part of the human body" (Bojko, 2013). Interestingly, as Holmqvist & Nystrom (2011) describe, saccades, and the stabilizing movements called *glissades* that punctuate the transition between saccades, can occur when there is no visual stimulus, even in the dark. These eye movements are most often associated with cognitive activity within the brain, and therefore it is possible to derive some knowledge about the brain's cognitive

processes by analyzing these movements (Biedert *et al.*, 2010), especially when the visual stimuli are being controlled in an experimental/laboratory setup. Somewhat counterintuitively though, the brain is currently understood to extract visual information only during relatively motionless periods in between saccades, the saccades themselves being momentary durations of time in which the brain is not 'encoding' any visual information (Poole & Ball, 2006). These motionless periods, when the eye is focusing on something in the visual field are called *fixations* (Bojko, 2013). These fixations can last from a few milliseconds to a few seconds (Holmqvist & Nyström, 2011), and may be considered to be the moments where the brain is actually investing attention to a particular element (or a closely bound set of elements) in the visual field. Holmqvist and Nystrom (2011) describe three other distinct types of movements of the eyes – *tremors* (small movements at a frequency of approximately 90Hz), *microsaccades* (movements that bring eyes back to their original position), and *drifts* (movements that cause the eyes to 'drift' from their original point of fixation). These three types of movements serve to avoid the saturation of the rods and cones on the retina, something that could lead to 'fading perception' (Biedert *et al.*, 2010). The multiple sequences of these movements and fixations strung together are referred to as *scanpaths*, and once visualized can offer incisive insights into the efficacy of an IIR system.

From the perspective of IIR research, and the broader level field of Human-Computer Interaction (HCI), fixations and eye movements have been associated with a variety of cognitive responses to features of information systems. Poole & Ball (2006) provide a condensed summary of how the different eye movement metrics can be interpreted while analyzing information systems with electronic display interfaces. Some

of the most salient relationships between eye movement metrics and associated cognitive activity are listed below:

i. a large number of overall fixations may indicate a less efficient search interface (Goldberg & Kotval, 1999),

ii. more fixations on a specific area may indicate that it is relatively more noticeable than other elements on the screen (Poole, Ball, & Phillips, 2005)

iii. longer fixations on specific areas may indicate either that the brain is facing difficulty in extracting information in that area, or that the concerned area is more engaging in some other way (Just & Carpenter, 1976)

iv. a higher saccade-to-fixation ration may indicate that the brain is expending more effort on searching for the information than on processing it (Goldberg & Kotval, 1999)

While research literature on eye tracking and cognition does provide a gamut of metrics similar to the ones mentioned above, the technology to measure them has significantly advanced since the earliest eye tracking research was carried out. Since a historical perspective on the evolution of eye tracking is not in the scope of this report, I now move to explain how eye trackers work, what are the different types of eye trackers, and how they can be used to gather data on the aforementioned metrics.

Chapter 6: What Are Eye Trackers, and How Do They Work?

In this section I briefly describe what an eye tracker is by describing the concepts behind its working, and the various types of eye trackers currently in use. Subsequently, I describe how a typical eye tracker hardware-software combination works by considering the example of a Tobii Pro TX300 eye tracker.

WHAT IS AN EYE TRACKER?

A minimalist, but generic, definition of an eye tracker is that it is a device that is used to determine where a person is looking. However, in the context of IIR research for information systems with electronic displays, it is more apt to define an eye tracker as a device that is used to study the visual behavior of a person while they are interacting with an electronic display or screen. Among the most important visual behaviors that most eye trackers track are fixations, saccades, and pupil dilation. Typically eye tracking equipment comprises of dedicated hardware to monitor the eyes, and software to process the data captured by the hardware.

HOW DO EYE TRACKERS WORK?

Modern eye trackers utilize reflections from the pupil and cornea to determine the exact position of the eye's gaze, or the 'point-of-regard' (Poole & Ball, 2006). The most common variety of eye trackers typically consist of source of near-infrared radiation, and a camera sensitive to the near-infrared spectrum. There are essentially two types of eye trackers currently in use in IIR research: *wearable* and *remote* eye trackers. Wearable eye trackers are typically worn by the participant on their head (Bojko, 2013), and do not have any other accompanying component other than the software running on a computer reading

the data from the wearable tracker. Remote eye trackers on the other hand typically consist of a separate hardware typically placed close to, and in front of the participant's body. Bojko (2013) offers a detailed comparison of the two types in her book. One of the main advantages of wearable eye trackers is that unlike remote eye trackers, they can be used to study the participant's gaze on whichever surface they choose to view (within the specified limits as ascribed by the underlying technology, and specifications). Remote eye trackers, on the other hand restrict researchers to capture particpants' gaze only on a fixed surface (most often a monitor or a display screen of some sort). This advantage of wearable eye trackers extends into allowing participants freedom of movement without affecting the accuracy of the tracking data, whereas remote eye trackers restrict the movements of the participants. I will restrict the discussion of eye tracker hardware to remote eye trackers in order to maintain the scope of this report.

As mentioned above, most remote eye trackers consist of a desktop computer monitor at the bottom (or top) of which are two pieces of imaging hardware: a source of near-infrared light, and a camera sensitive to that range of wavelengths. In order to precisely locate the position of the eye, infrared light from the source is directed in the general area surrounding the eyes resulting in strong reflections. This is where our knowledge of the structure of the human eye comes in to the picture. As I described in the section on the human eye, the external surface of the eye just over the iris is called the cornea, and the circular region enclosed by the iris is called the pupil. Now, when infrared light rays fall on the eye, at least two reflections strong enough to be detected by the infrared camera are created. One of the reflections is created as a result of the cornea reflection the incident infrared light (called corneal reflection), and the other is created by the light reflected off the retina and let back out through the pupil (pupil reflection). These

two types of reflections have distinct properties that allow the infrared camera embedded in the eye tracker deduce the precise location of the eye's gaze. Using basic trigonometric calculations, and in concert with information about the location of calibrating elements on the screen, the embedded software can precisely figure out where the eye is looking.

One important thing to note about this eye tracking method is that it can only capture the foveal vision, and does not offer any insight into the peripheral vision of the person whose eyes are being tracked (Bojko, 2013).

DATA COLLECTION WITH EYE TRACKERS

Two of the most important considerations about tracking the eye's gaze are spatial precision, and temporal granularity. Spatial precision is typically achieved by ensuring that the eye tracking hardware is concealed with the actual display into a single monitor unit, and by ensuring that proper calibration routines are followed before any data collection is done. Another aspect, which was listed as a limitation of remote eye trackers, that helps ensure spatial precision or accuracy is that the position of the particpant's head should not change by more than a specified number of degrees with reference to the eye tracker hardware. Generally, this freedom of movement is restricted to a 'head box', which is typically defined as a region 12-17 inches wide, 7-9 inches high, and 8-12 inches deep (Bojko, 2013).

Temporal granularity, on the other hand refers to the idea that how frequently is the position of gaze recorded during an experiment. This duration is determined by the *sampling rate* of the eye tracker hardware, and measures the number of times the eye tracker registers the participant's gaze per second (measured in Hz) (Bojko, 2013). A

higher sampling rate entails more precise measurement of fixation durations, and accurate registration of scanpaths.

The goal of ensuring the above aspects of accuracy is to get highly reliable data about the various eye tracking metrics I presented earlier in this report. It is important to revisit some of those metrics before I move to the section on experiment design and using eye tracking software. The metrics typically of most interest to IIR researchers are the following:

i. **Gaze points and fixations:** gaze points are the most basic unit of measuring the location of the eye's position. Each sample collected by the eye tracker is essentially a gaze point, several of which are aggregated into fixations depending on the duration for which the gaze points were approximately fixed on a specific object on the screen (iMotions, 2016).

ii. **Saccades:** These are the movements eyes make in between fixations.

iii. **Scanpaths:** The overall paths formed by the sequences of fixations and saccades.

iv. **Heat Maps:** These are visual representations of fixation data typically overlaid on the visual stimuli used during the data collection. These heat maps borrow from thermal heat maps, whereby regions that received most fixations are rendered red, and those that received relatively fewer fixations are rendered blue. These maps are great tools to visualize data about what captures participants' attention in the visual stimulus (e.g., a webpage, or an image).

Apart from the metrics mentioned above, most eye tracking software programs also allow researchers to design Areas of Interest (AOIs), or regions within the larger visual stimulus that are of particular research interest. For example, in the case of studying participant interaction with search result pages, one can instruct the eye tracking software to capture fixation or saccade data only for the search bar and the web page titles in list of results.

EYE TRACKERS AND IIR

When it comes to relevance judgment in web search interfaces, Granka, Joachims, & Gay (2004) point out at least two distinct advantages of using eye tracking: eye tracking has the potential to get researchers improved metrics about specific targets in search interfaces, and it helps researchers collect and interpret implicit feedback about users' interactions with a results page. Such implicit feedback, as I have also described previously, is difficult to obtain through either of the two variants of the 'think aloud' protocols, or even video recordings of users interacting with the systems. Granka *et al.* (2004) further mention that some of the most key metrics among this implicit feedback are what abstracts or synopses a user reads while browsing through search results, for how long, and in what order. Cutrell & Guan (2007) also show promise in eye tracking as a means to answer research questions about whether at all users read the search result abstracts, do they inspect metadata like URLs like the expert searchers do, or does the context of the search matter?

On the surface much IIR or IR research seems to be focused on digital systems, and digital documents. But Bucher & Schumacher (2006) offer an interesting, and somewhat comparative, analysis of how the differences between print and online news interacts with reader attention and selectivity. At the same time the comparison also raises a question as

to whether mere salience-based hierarchy of content presentation in a print news paper is responsible for soliciting selective attention (Bucher & Schumacher, 2006). I believe there is a lot of import from such a comparative analysis, since a search interface when presenting results must also similarly interact with user attention and selectivity.

One of the simplest ways that search engine designers can gauge users' relevance judgment heuristics is to monitor their click behavior – which links they clicked on, which links did they hover the mouse pointer on, and how far down did they scroll before finding a link they found relevant enough to click on. But this simplistic approach falls short as soon as the user browses away from the search results page, and onto a document/webpage referred to by the results page. Each time a user browses away to inspect a document and decides to come back to the results page to look at more results, there is an opportunity lost as to understand what caused the user to come back and explore more. Why did the user come back to the search results? What did the user not find on the page they browsed? A relevance inference strategy that is purely based on mouse (or keyboard) interactions can be especially limiting when the browsed document/webpage does not have any other hyperlinks. This is precisely the kind of void eye tracking has promised to fulfill in the recent years. Specifically, eye tracking can enable the setting up of laboratory conditions whereby the interaction between users' attentional resources and the contents of web pages (other than just the results pages themselves) can be recorded and inferred from.

Buscher, Dengel, & van Elst (2008) offer a particularly salient example of using eye tracking to gauge whether a user is merely skimming a given document, or actively engaged in reading it. The contrast between skimming and thorough reading is also an integral part of Aula, Majaranta, & Räihä's (2005) study to differentiate different 'styles' of result evaluation where they ultimately classify users as *economic*, and *exhaustive*

evaluators. As I mentioned previously about the hierarchy of content presentation as a probable interactive factor in content selection and attentional investment, it is vital for search engines to not just present the 'right' abstracts, but also to emphasize the 'right' words or phrases in those abstracts. Such emphasis is typically achieved by increasing the font weight of *predicted* relevant words, and obviously by *ranking* results. Such exercises in emphasizing content are not the sole prerogative of search engine designers, but have been practiced by content creators across genres. This may include bloggers who emphasize words, strategically hyperlink their other articles and advertisements, and video streaming services like YouTube or Netflix which strategically place 'suggestions' and user ratings to influence or solicit attention.

Eye tracking offers a window of sorts into users' brains, or minds if we consider their values and predispositions, and offers great promise to first analyze digital information layouts, and therefore, to improve the design of these digital systems. In other words, there is a world of possibilities to be uncovered by eye tracking when it comes to knowing how users, or specifically their brains, react to web-based search interfaces and documents. Bringing the focus sharply on to the information presented on the Web, I now explore how the promises held by eye tracking can be realized to get some tangible insights about web interfaces.

Chapter 7: Experiment Design in Eye Tracking Based IIR Research

In general, most experimental research employs one or more of the following three basic types of research approaches:

i. Exploratory research: This type of research is typically conducted when not much is known about the phenomenon under investigation (Kelly, 2009). In addition, exploratory research employs a variety of methods, some of which may be unstructured.

ii. Descriptive research: This type of research is generally conducted with an intent to describe in more detail phenomena which have been explored before, and about which some understanding has already been developed.

iii. Explanatory research: This type of research typically tries to establish relationships between the different variables involved in the phenomenon under investigation. Explanatory research generally uses more structured methods, and mathematical models to establish relationships between variables. Such models not only serve to explain the phenomenon under investigation, but may also offer predictive capabilities, *i.e.* predicting how the phenomenon could unravel given a specific set of conditions are applied or are true.

Most research ultimately leads to an evaluation phase, where results are evaluated with the goal of applying the results to a specific purpose. In IIR the purposes typically range from understanding and improving the interactions between users and IR systems, and improving the presentation of information on webpages. The latter, *i.e.*, the study of presentation of information on webpages forms the focus of the rest of this report.

Whereas the classification of research above refers to the philosophy of understanding a given phenomenon, there is another classification scheme that fits well with the topic of IIR. According to this classification, IIR research can be classified into two broad categories: *experiments* and *evaluations*. Experiments typically involve the study of the relationship between independent variables, and one or more dependent variables. In addition, experiments involve the comparison of two or more conditions – with at least one 'treatment' condition where an independent variable has been operationalized and manipulated, and at least one 'control' condition where no modification is done. Such research typically results in more research questions and hypotheses which are tested and either retained or rejected. In an IIR context a typical experiment could be the study of a user interface (e.g., a search result page) where different elements of the interface (independent variables) are manipulated, and the effects of those manipulations (e.g., perceived relevance) measured.

Evaluations, on the other hand are conducted to assess qualitative aspects of the system under study. These qualitative aspects could, for example, include the "goodness" of a system or an interface (Kelly, 2009), or the usability of a web application. Oddly, one of the best ways to describe evaluation is to use the concept of experimentation itself. Kelly (2009) describes evaluation in this way by asserting, "[O]ne can conduct an evaluation without conducting an experiment and *vice versa*", and defines two types of evaluation that are relevant to IIR: *formative evaluation* as evaluation that is conducted during the design of a system, and *summative evaluation* as evaluation that is done to assess the value of a mature system (Kelly, 2009).

In the upcoming sections, I will conceptually borrow from both the above mentioned classifications of IIR research, and apply them to experiment design specifically targeted towards the study of scrollable web pages.

COMPONENTS OF AN EYE TRACKING STUDY

Borrowing from Bojko's (2013) general questions that can be answered using eye tracking experiments, here are the typical attraction-related questions that can be asked about webpages in an IIR context:

i. Where should important content be placed on a webpage?
ii. How should important content be designed so as to attract and capture user attention?
iii. What overall page design sustains the users' interest the longest?
iv. Does the new design of a key element fare worse or better than the existing design in terms of grabbing attention?

Similarly, the following performance-related questions could be asked about webpages:

i. Does a redesigned webpage make it easier for users to find the information they are looking for?
ii. Does the inclusion of more extra white space on the page reduce the number of fixations before a user locates the relevant piece of information?

In order to answer the above questions, IIR researchers typically employ the following components in their experiment designs:

i. Visual stimuli: Visual stimuli act as inputs to the human visual system that trigger cognitive activity within the brain. A webpage, for example, provides a stimulus to the brain by virtue of having characteristics like the type, color, size, and hierarchy of text, the placement of images, embedded animations, *etc.*

ii. Flow control: A mechanism to control the flow of the various portions of the experiment. These portions include the eye tracker calibration routines, presentation of visual and/or other stimuli.

iii. Data collection: A software program that records the data from the eye tracker and combines it with the visual stimuli to generate visualizations like scanpath plots and heat maps.

As outlined in the introduction to this report, I now describe the process of conducting eye tracking experiments using the software program EventIDE.

Chapter 8: Designing and Conducting Eye Tracking Experiments with EventIDE

EventIDE is a software program created by Okazolab Ltd. designed to conduct behavioral, neuroscientific studies and usability tests. The company describes EventIDE as a ‘stimulus presentation software’, and as a visual programming platform for designing and conducting psychological experiments. The range of stimuli that can be used in an EventIDE experiment include video clips, audio clips, pictures, virtual reality based stimuli, 3D scenes, live websites, and other dynamically generated stimuli. When it comes to data collection capabilities, EventIDE can capture eye tracking data, electroencephalogram (EEG) and magnetoencephalogram (MEG) data, face tracking data, and can recognize emotions, motion, handwriting, and gestures (CITE: Okazolab). To stay within the scope of this report, I will restrict the following discussion to the eye tracking capabilities of EventIDE.

The eye tracking support in EventIDE includes real-time saccade and fixation detection, live analysis with areas of interest (AOIs), heatmaps, and scanpaths, and support for scrollable websites. In terms of hardware capabilities, EventIDE supports eye trackers with a sampling rate of up to 2000Hz, fully customizable calibration routines, and most popular eye tracking hardware including Tobii, EyeLink, and EyeTribe among others (CITE: Okazolab).

While there are many other software suites available for conducting similar studies (e.g., iMotions, and Morae), EventIDE offers the ability to ‘inject’ software scripts into an experiment design. This ability to customize the flow of an experiment using a simplified coding model confers EventIDE a distinct advantage, as it allows researchers to create experiment designs that can modify or adapt their behavior based on the values of selected

programming variables. For example, these variables could be used to monitor fixation duration, or whether a participant's gaze crosses a predefined boundary, to take specific actions like modifying the stimulus to be presented next, or terminating or restarting an experimental step if a certain condition has been met. EventIDE renders the task of coding into a simple exercise where the experimenter is only required to write simple statements in C# to create powerful and adaptive experiments.

When it comes to studying IIR within the context of webpages, EventIDE offers a distinct advantage by allowing a web browser instance to run *within* an experiment. This is one of the most powerful features of EventIDE as it allows greater control of the browser (monitoring URLs, injecting and executing custom Javascript code, creating dynamically defined AOIs), and thereby also improves the real-time performance for browser-based experiments. In the following sections I will describe the basic components of an EventIDE based experiment design and workflow.

BASIC COMPONENTS OF AN EVENTIDE EXPERIMENT DESIGN

At the topmost level, an EventIDE experiment comprises of the following components:

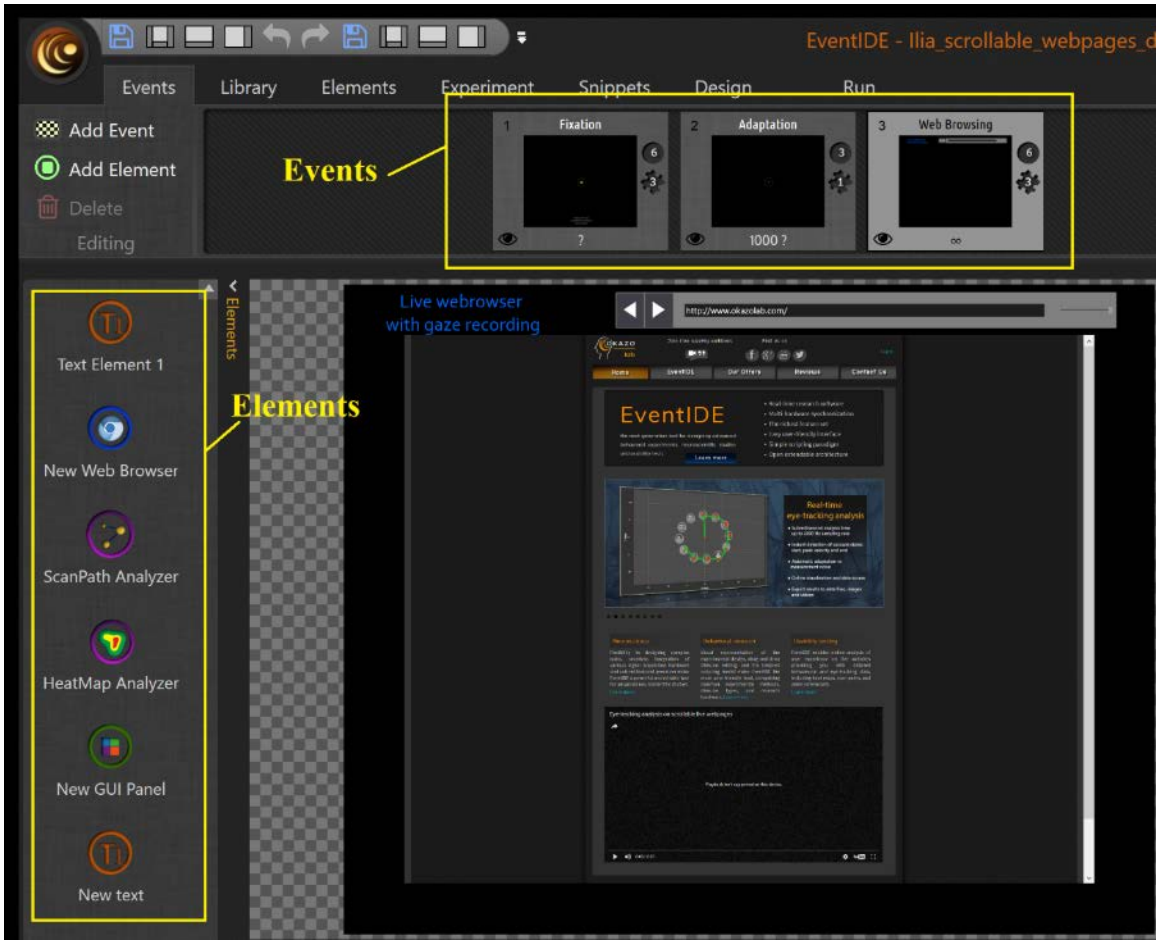

Figure 2: Events and Elements in EventIDE

i. **Events:** In the most abstract sense, an experiment in EventIDE comprises of a sequence of ‘events’, which the EventIDE documentation describes as being equivalent to ‘slides’ in a presentation. Technically, though, an event is a period of time that corresponds to a specific logical state in an experiment. This period is bounded by what are called *onset* and *offset* times. EventIDE also allows experiment designers to organize events in layers, *i.e.* a hierarchical organization of events, sub-events, and so on. Within each layer, however, only one event can be active at a given time,

but events belonging to different layers can be active at the same time (Okazolab, 2016). Each event also has a set of properties that can be accessed or modified during an experiment.

ii. **Elements:** Elements are containers of both – sources of experimental inputs (stimuli), as well as data output and data monitoring objects. Each event contains one or more of these elements. Each element has an extensive set of properties that can be read or manipulated during the course of an experiment.

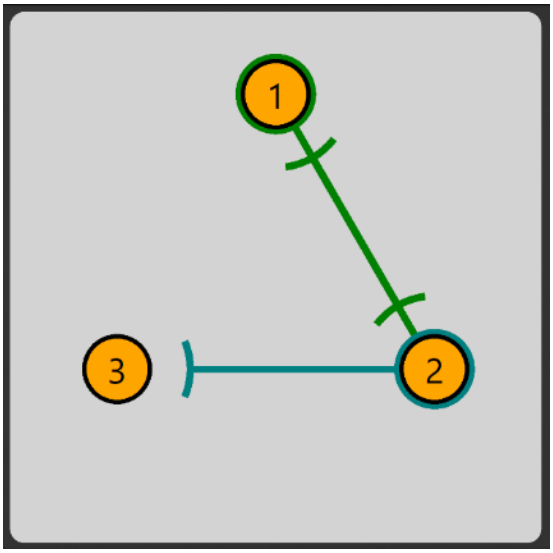

Figure 3: Flow Routing in EventIDE

iii. **Flow routes:** Flow routes define the sequence of events that will be executed during the experiment. One of the main strengths of EventIDE is its capability to conditionally route the flow from one event to the other. The conditions that can be used to dynamically select which event to execute next are listed below.

a. Time-gating: This allows experiment designers to define the maximum duration an ancestor (or previous) event should last, and the next event to be executed is selected based on the actual duration of the ancestor event. In Figure 3 above, the route between event 2 and 3 is directed from event 2 to event 3, and the blue color indicates that it is a time-gated route.

b. Logical expressions: Experiment designers can define custom logical expressions in C# in order to automate the selection of the next event to be executed. Logical routes are depicted with green color. In Figure 3, events 1 and 2 are connected by two routes – one in each direction. In this scenario values of the respective logical expressions determine whether events 1 and 2 keep alternating with each other, or whether event 2 ultimately gets to transition to event 3.

c. Mix route: This option allows experiment designers to combine both time-gating and logical expressions using AND/OR operations to define conditional routing of events. When time-gating and logical expressions are combined using the AND operator, the mix route is depicted with a purple colored connector. When the two conditions are combined using an OR operator, the mix route is depicted with a blue-green colored connector.

iv. **Stimuli:** Each EventIDE experiment contains an entire library of stimuli.. This library is referred to as the ‘Material Library’, and consists of all types

of stimuli such as pictures, patterns, text, object animations, videos, sounds, *etc.* (Okazolab, 2016) that are used in a given experiment.

Since stimuli are the most important components of any experiment in IIR research, let us now look at how EventIDE allows presentation and management of stimuli within experiments.

PRESENTING STIMULI

As mentioned previously, elements are the basic building blocks that together define an event. Each element is a container that encapsulates either a single stimulus, or a list of stimuli. In addition, elements can be ‘stacked’ or layered on top of each other as shown in Figure 4.

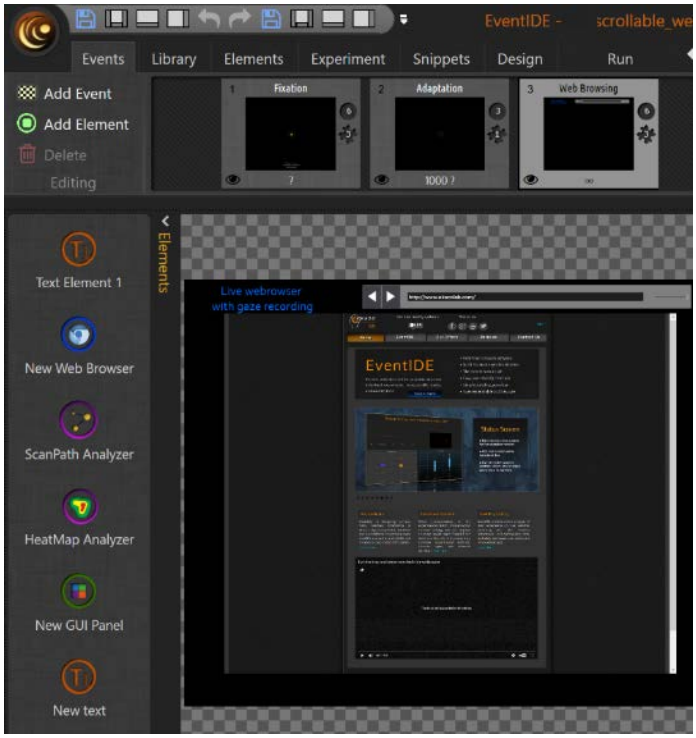

Figure 4: Elements in the left panel

The left-most column (outlined in yellow) in the figure contains the elements belonging to event number 3, named “Web Browsing”. As can be seen in Figure 4, the element called “Text Element 1” is at the top of the column; in EventIDE, this implies that it is *first* in the stacking order, or at the *bottom* of the stack. This arrangement causes the contents of the element to be invisible because the “New Web Browser” element is stacked on *top* of it. However, as shown in Figure 5, when the “Text Element 1” is placed *below* the “New Web Browser” element in the stacking order, its contents (the text string “Eye Tracking”) is seen overlaid on top of the web browser element.Stimuli can be added to an experiment by first adding an element appropriate to the type of stimulus.

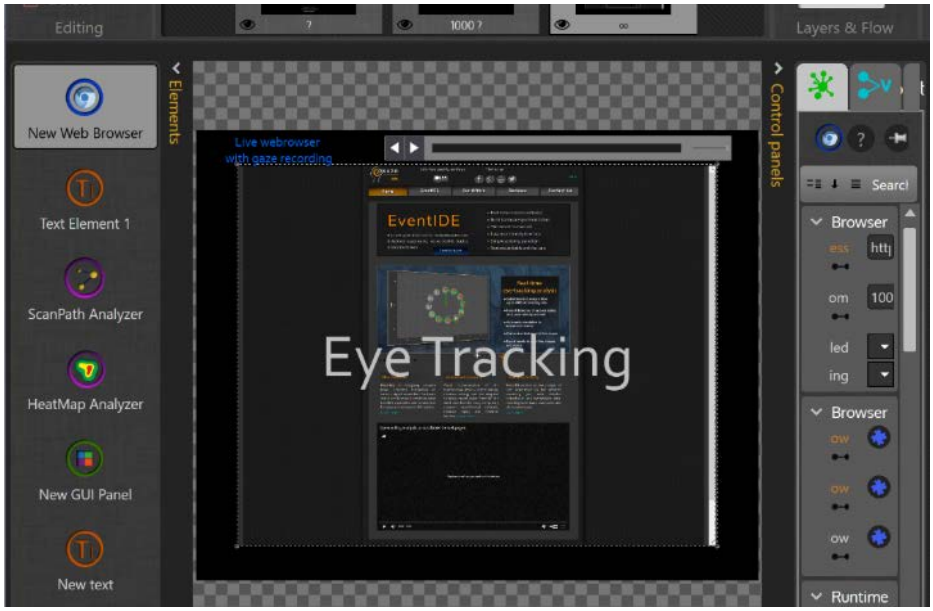

Figure 5: Effect of ordering elements on how elements are displayed.

For example, in order to display a picture as stimulus, one needs to add a ‘Renderer’ element to the event. This element can be added by clicking on the “Add Element” icon in the “Events” ribbon menu at the top of the interface. This process is shown in Figure 6 (next page).

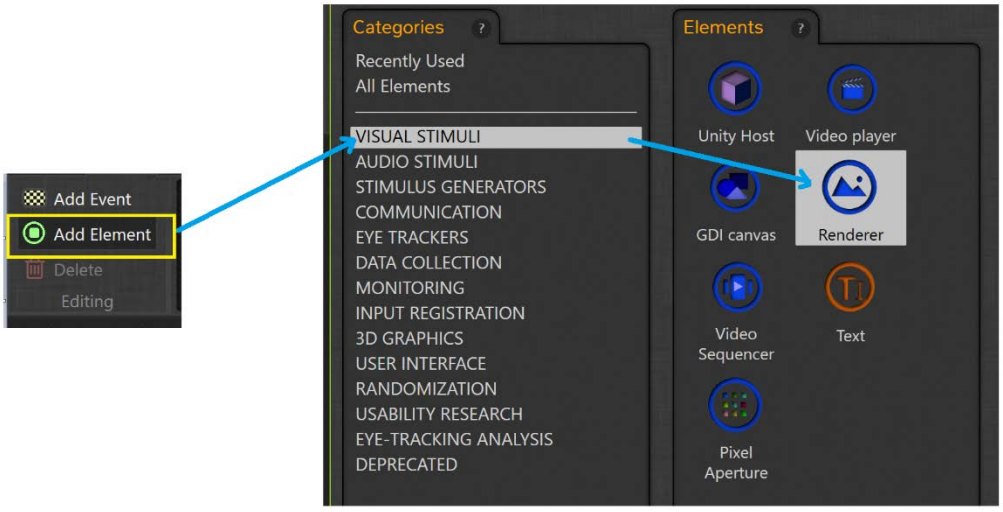

Figure 6: Adding a visual stimulus element to an event

Once the renderer element has been added, one can add a picture (or another visual stimulus) to it by dragging and dropping the picture to the element. In a similar way, multiple pictures or other visual stimuli can be added to the renderer element, and the ordering of the stimuli within the renderer element can be changed.

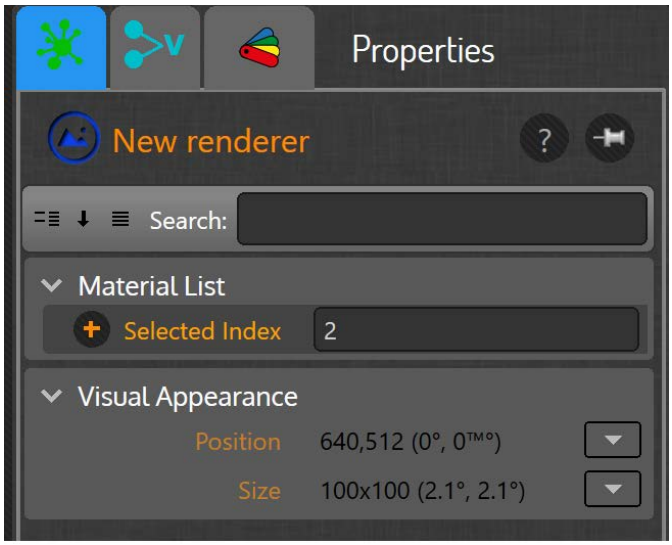

Figure 7: Selecting the index of the stimulus to be used in a trial

Each stimulus in such a list is assigned an ‘index’, and the index (and therefore the stimulus) to be selected can be modified in the ‘Properties’ pane to the right (shown in Figure 7). This is where EventIDE’s scripting feature bears out its advantage. By creating a ‘proxy variable’ (in other words an alias) that refers to the ‘Selected Index’ property of the renderer element, one can easily select a different (or the same) stimulus from the list each time the parent event is executed.

Similarly, other elements can be added to an event, and each event can be loaded with a stimulus appropriate to that element. However, not all elements can be loaded with new stimuli. For example, the “Web Browser” element serves as a stimulus by itself as it

creates a new web browser instance. In a similar way, each element has its own properties (accessible via the 'Properties' pane as shown in Figure 7) and restrictions. Most properties associated with elements can be made accessible by creating a proxy variable by clicking the icon next to the property name (outlined with yellow in Figure 8 for the 'Web Address' property of the Web Browser element).

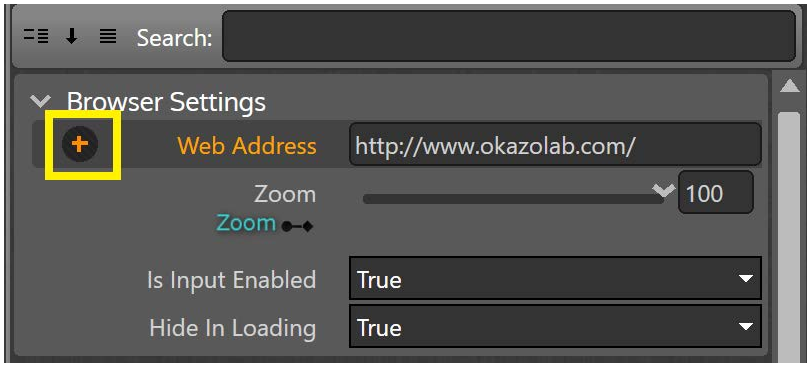

Figure 8: Creating a proxy variable for a property

Proxy variables so created can be used to create powerful and dynamically configurable assignments. Having briefly described the basic components of experiments in EventIDE, and having introduced the concept of proxy variables, I will describe the scripting functionality built into EventIDE in the following sections.

SCRIPTING IN EVENTIDE

Behind the user interface, and graphic UI elements of EventIDE lies a powerful programming platform that supports programming in C#. The most important feature of programming in EventIDE, however, is that it does not require researchers to write functions, procedure calls, or other complex programming constructs in order to customize

the experiment design. Instead of conventional programming paradigms (e.g., procedural programming) EventIDE facilitates an event-driven programming approach. The logical routing flows and element properties constitute the main control flow of an experiment. But writing simple scripting statements attached to specific stages of an experimental flow one can create ‘event handlers’ that respond to specific events in the course of an experiment.

Much customization can be achieved by reading and modifying properties via associated proxy variables from within the scripts described above. These user-defined scripts are placed in ‘snippets’, which are associated with specific stages of an experiment. The EventIDE documentation (Okazolab, 2016) describes the following kinds of snippets.

Snippets at the Experiment Object Level

i. **Header:** This snippet contains declarations of global variables and functions which can be accessible from all the other snippets across the experiment.

ii. **Status Screen:** This snippet is used to create or render a ‘status screen’ (for use in a separate monitoring or observation room). The elements of the status screen are defined using the Extensible Application Markup Language, or XAML.

iii. **Start:** This snippet is executed at the beginning of each experiment, and is executed only once. Therefore it is used for various initialization actions and runtime variable assignments.

iv. **Control Loop:** This snippet contains code meant to be executed repeatedly during the whole duration of the experiment. The EventIDE documentation advises that this snippet should contain only ‘relatively short’ code that

executes quickly. This requirement has been specified to ensure accurate timing in an experiment.

v. **End:** This snippet is called at the end of the experiment, and is intended for the various ‘finalization’ actions.

Snippets at the Event Object Level

i. **Route Conditions:** This snippet is intended to define the conditions to control the flow routing that I described previously in this report. The conditions include time durations for time-gating based route flows, and logical expressions for the other conditional route flows. The corresponding logical expressions are evaluated at runtime until its result is true.

ii. **Rendering:** This snippet is called each time the visual content of the owner event is changed.

iii. **Before Onset:** The code in this snippet is called just before the onset, or start, of the owner event.

iv. **After Onset:** This snippet is executed immediately after the onset, or start, of the owner event. This snippet is intended to execute all actions that have to be synchronized with the owner event.

v. **Control Loop:** This snippet is similar to the control loop snippet at the experiment object level, but is instead intended for actions that need to be executed repeatedly while an event is active. Also, just as in the case of the experiment object level control loop, study designers should avoid placing long and complex pieces of code in this snippet as that could affect the accuracy of some time-critical functions.

vi. **Before Offset:** This snippet is executed when an event flow is switching or transitioning to another event (which could be the same event in case of a cyclical flow routing condition). It is important to note that this snippet is called in a pair with the 'Before Onset' snippet of the next event in the flow route.

Snippets at the Element Object Level

At this level there is currently only one type of snippet, the 'Triggered' snippet. This snippet is executed each time the status of the owner element is changed and has to be reported. The status change could, for example, include detection of a button press, or arrival of data on a data port. This also implies that this snippet is called 'asynchronously', *i.e.* its execution is contingent on the change of status of an element.

Accessing and Editing the Snippets

The various snippets described above can be accessed via icons dedicated to each type of snippet at the bottom of the main EventIDE window – the 'Snippets Panel'.

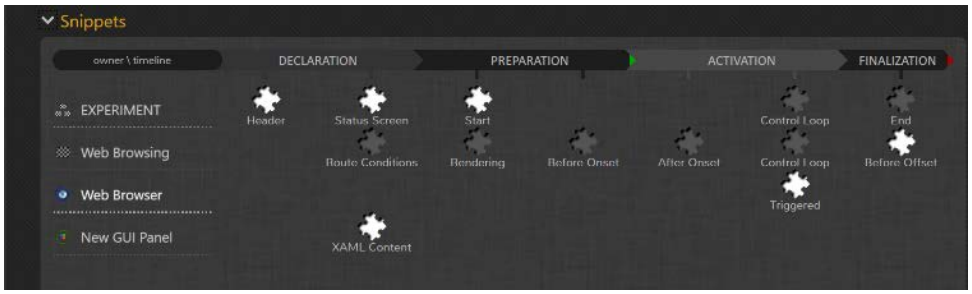

Figure 9: The Snippets Panel

As shown in Figure 9, the different types of snippets appear in the row corresponding to the level of the owner object. For instance, the snippets owned by the experiment object (e.g., ‘Header’, ‘Status Screen’, *etc.*) are located in the row titled ‘Experiment’. The icons with a solid white fill indicate that the linked snippet has some code in it, whereas those with a dark grey fill indicate that the linked snippet is empty.

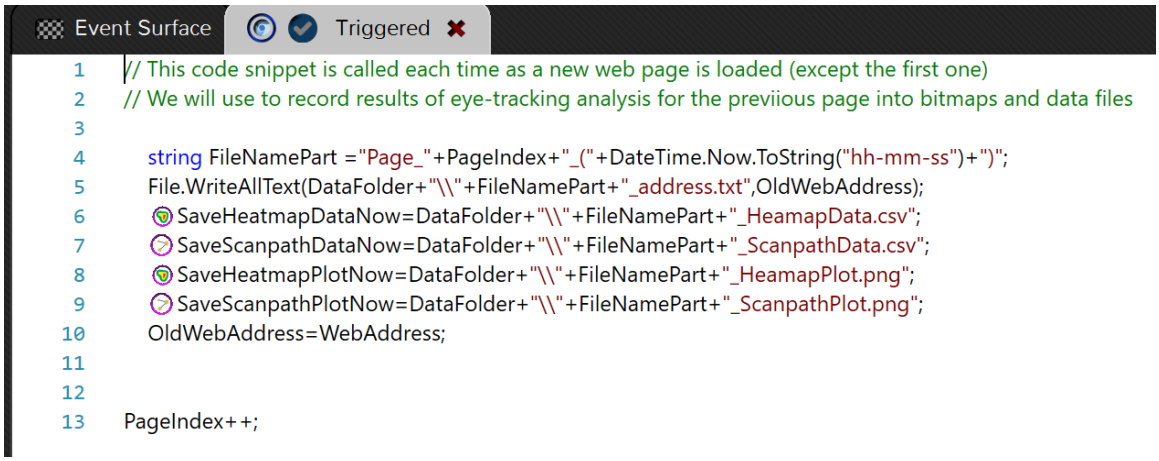

Figure 10: The Code Editing Window

Clicking on any one of the snippet icons opens up an editor window (Figure 10) where one can write custom code in standard C# syntax.

CREATING AN EVENTIDE EXPERIMENT TO STUDY A SEARCH RESULTS PAGE

While the EventIDE documentation does not provide step-by-step instructions about how to create an experiment to study the usability of webpages using eye tracking, it offers numerous templates to base typical experiment designs on. I will explain one such template design below.

1. Referring to Figure 11, click the EventIDE menu icon at the top-left, and click on ‘Demos’ from the menu. Then browse through the template categories in the ‘Online experiment gallery’, and navigate to ‘Usability Research’. From that category, click on the experiment template called ‘Usability analysis on scrollable webpages’. This loads an experiment with all the basic elements needed to run an eye tracking study on scrollable web pages.

Figure 11: Loading the Built-in Usability Research Demo Experiment

2. The way the demo is set up, it uses an ‘EyeTribe’ eye tracker by default. In our case, I used the Tobii eye tracker in the laboratory. To remove the EyeTribe tracker, and instead add your own tracker follow these steps:
    a. Navigate to the Fixation event, and right click on the EyeTribe element. As shown in the Figure 12, select the option “Delete element”. This will remove the EyeTribe element.

b. Now click the “Add element” button, select “Eye Trackers” from the Categories pane, and then choose the eye tracker element appropriate to your system. In my case, I choose the “Tobii Tracker” element.

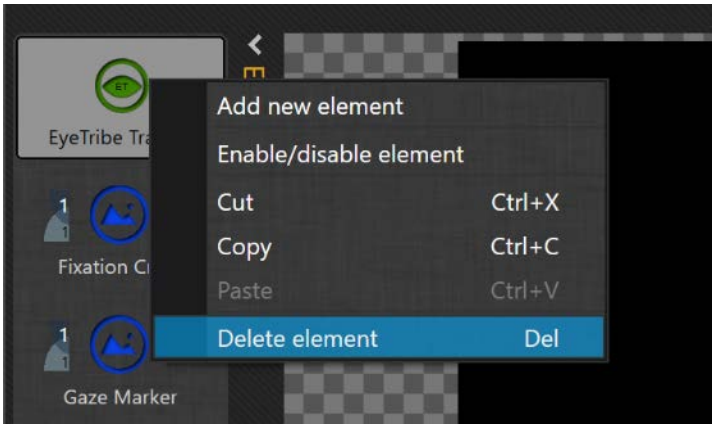

Figure 12: Delete the EyeTribe Tracker element

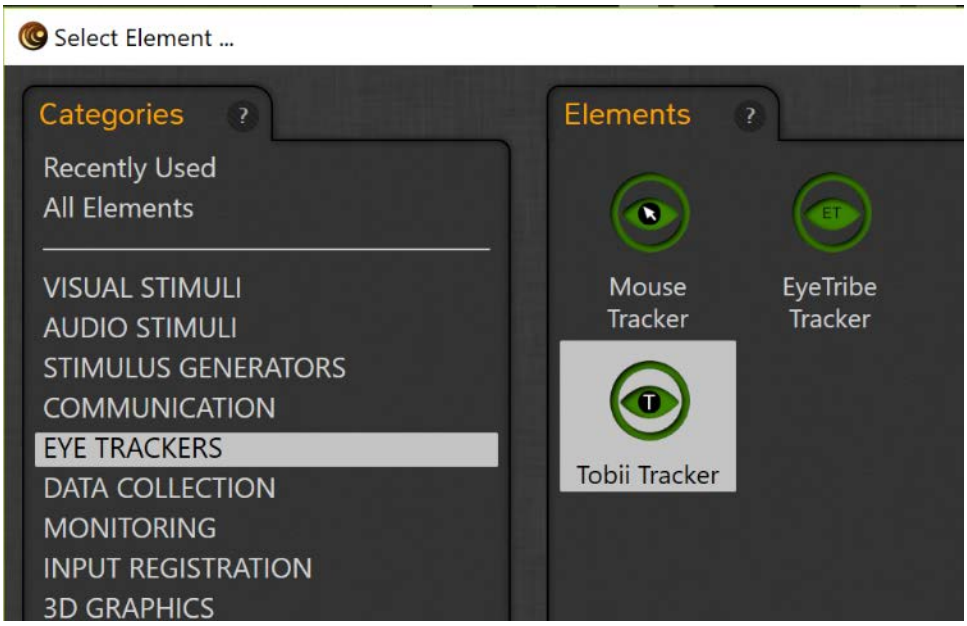

Figure 13: Select the Tobii Tracker element

c. Create a proxy-variable for the property “Recalibrate Now” in the Tobii Element, and name it “RecalibrateNow”.

3. The other elements in the Fixation event – Fixation Cross, Gaze Marker, Space, and Fixation Area – are what aid the participant in locating the position where to focus their gaze. Additionally, the Fixation Area defines a circular region of interest (ROI) around the Fixation Cross and is used to determine how stable the gaze is within the circular region. The proxy variable ‘FixationStarted” associated with the property “Is Triggered” is used to signal to EventIDE to route the experiment flow to the second event, “Adaptation”.
4. In the Adaptation event, we see the “New Velocity Threshold Estimator” element, which uses its existing knowledge of saccadic movements to estimate which eye movements could be reasonably interpreted as saccades for the current participant. The Fixation Cross is similar to the similarly named element in the Fixation event. Its purpose is to keep the participant’s gaze focused in a specific area. It’s position is kept the same as in the Fixation Cross element in the Fixation event.
5. Similarly, we have a Fixation Area element at the bottom. In this case, however, the “Is Triggered” property is used to define a new proxy-variable called “IsFixationBroken”. If the participant’s gaze ventures out of the ROI defined by the Fixation Area element, this proxy variable will be set to False. Now in case it is False, the experimental flow would route the control back to the first event, *i.e.*, the Fixation event. If you closely notice the bottom of the Adaptation event icon, it shows that this event has a minimum duration of 1000ms. What this implies is that at the end of 1000ms, the value of IsFixationBroken will be tested, and if it is False, the experiment will move back to the first event.

6. Now let us take a look at the final event, “Web Browsing”. As shown in Figure 14, it has 5 elements: New Web Browser, Web Scanpath Analyzer, Web Heatmap Analyzer, New GUI Panel, and New Text. The New Web Browser element is what creates an instance of a Google Chrome based browser when the event executes.

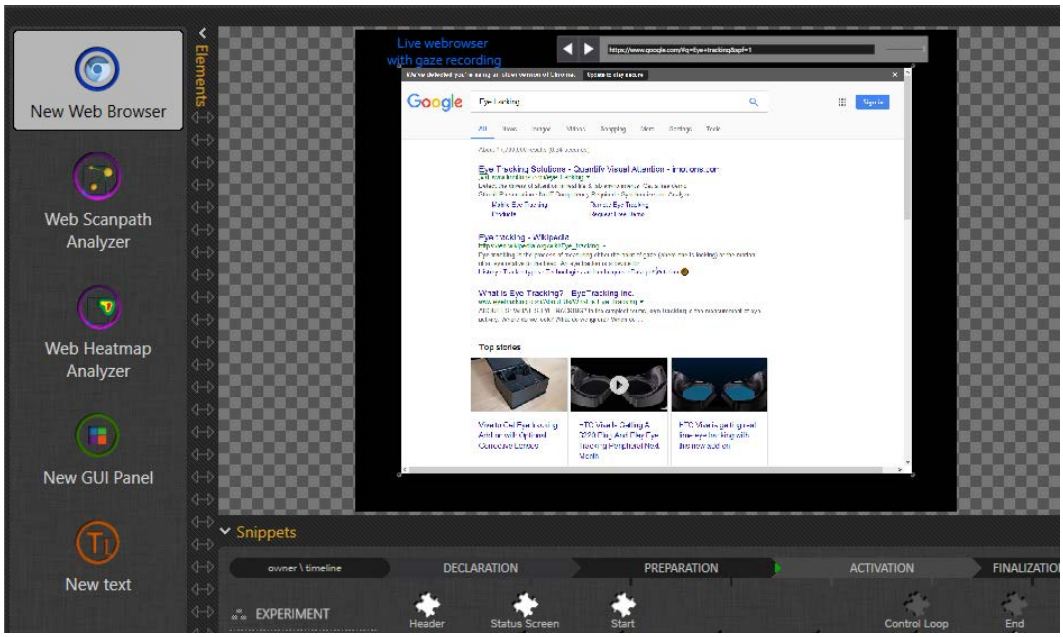

Figure 14: The Elements in the “Web Browsing” element

a. The ‘Web Address’ property of this element is where the experiment designer can specify the URL where the browser would navigate to once the experiment is conducted. The participant will interact with the page rendered in this browser.
b. The Web Scanpath Analyzer element captures the saccade data, and generates scanpath plots as .PNG images.

c. The Web Heatmap Analyzer similarly analyzes the participant's gaze, and generates fixation duration based heatmap data and .PNG images.

In order to run this experiment, navigate to the 'Run' ribbon menu at the top. Make sure the running mode in the "Run Experiment" section of this menu displays "GUI Mode". Then click on the "GUI Mode" button. This will launch the experiment and the events will be executed as per the flow-routing defined in the experiment. Once the participant ensures a long-enough fixation in the Fixation Area ROI defined around the Fixation Cross, the flow will route to event 3, "Web Browsing". Let the participant interact with the web page (pointing to www.google.com in our case).

Shown below are the scanpath (Figure 15) and heatmap (Figure 16) plots for a sample run of the experiment.

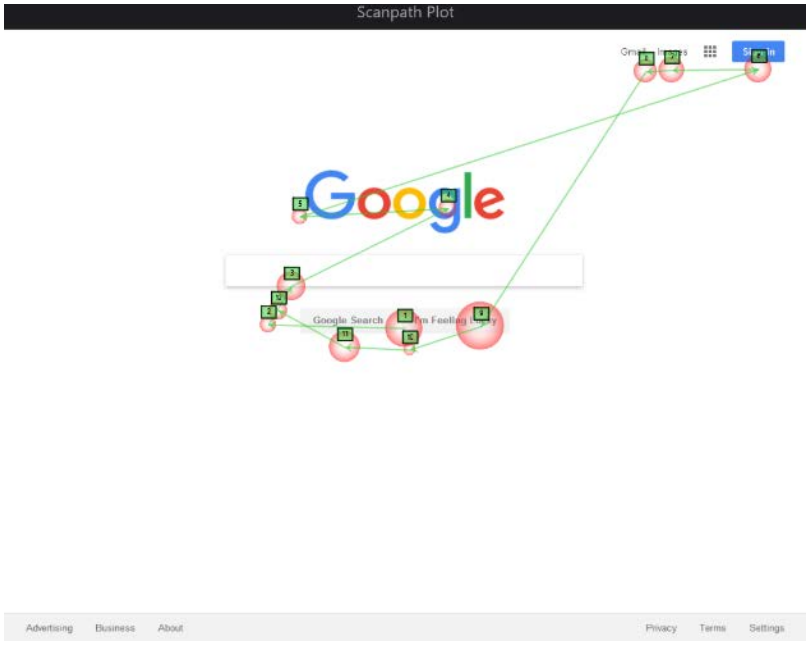

Figure 15: Scanpath Plots from the Demo Experiment

As can be seen in Figure 15, the scanpaths are made up of shaded circular shapes and straight lines. The diameter of a circle denotes the duration for which the participant's eye was fixated at a particular location. The numbers within each circle indicate the order in which each location was gazed at. Figure 16, however, shows a heatmap plot where the duration of fixation at a specific location is rendered as a colored cloud. The color ranges from blue (least duration of fixation), to red (maximum fixation duration), with intermediate shades showing intermediate levels of fixations.

Figure 16: Heatmap plot for the Demo Experiment Run

As can be seen from the above heatmap, and the scanpath plots that a considerable amount of attention was focused on Google's "I'm Feeling Lucky" button, the initial portion of the search bar, and the login button and menu items on the top right. While these plots could be very different for different users, they show that in the case of this experimental run, the participant's eyes were primarily fixated on the elements as described above. Apart from plotting the gaze data in this fashion, EventIDE also allows one to export the data in a CSV format.

Chapter 9: Concluding Remarks

In this report I have described the role of eye tracking in IIR research by situating IIR in its larger context. That context includes the field of IR and the psychological implications of human vision. As a concluding remark, it is appropriate to say that the field of IIR primarily deals with modern visual interfaces that people use to search for and browse information online (or locally on their computing devices). Whilst IR research is highly focused towards understanding the systems that take information queries, investigating the kind of results the systems output, and measuring how users judge the relevance of those results. IIR, in contrast, although not completely divorced from IR, factors the user's psychology and provides insight into how they judge relevance. This relevance is inferred not just for the content or synopses of, say, search results, but for other purely visual features of interfaces like the HTML layout, colors, amount of white space, *etc.*

In the second half of the report I demonstrated a software program called EventIDE that allows IIR researchers to conduct a wide variety of behavioral experiments. In keeping with the scope of this report, I focused on eye tracking. EventIDE offers experiment designers the ability to write code snippets in an easy to understand C# syntax to customize the flow of their experiment. Whereas purely Graphical User Interface (GUI) based software allow easy setup of experiments, the coding feature built in to EventIDE expands the amount of control researchers get in designing experiments. Using the specific demo experiment built in to EventIDE I demonstrated how it can be used to study the efficacy and usability of a web interface like that of Google.com.

In conclusion, I hope this report provides a generic introduction to the topic of IIR, and a practical insight into how to conduct experiments. I strongly encourage readers to

explore the topic in more depth using the references mentioned at the end of this report, and explore EventIDE for conducting experiments.

Bibliography

Aula, A., Majaranta, P., & Räihä, K.-J. (2005). Eye-tracking reveals the personal styles for search result evaluation. *Human-Computer Interaction-INTERACT 2005*, 1058–1061.

Balatsoukas, P., & Ruthven, I. (2010). The use of relevance criteria during predictive judgment: An eye tracking approach. *Proceedings of the American Society for Information Science and Technology*, *47*(1), 1–10. https://doi.org/10.1002/meet.14504701145

Biedert, R., Buscher, G., & Dengel, A. (2010). The eyeBook – Using Eye Tracking to Enhance the Reading Experience. *Informatik-Spektrum*, *33*(3), 272–281. https://doi.org/10.1007/s00287-009-0381-2

Bojko, A. (2013). *Eye Tracking the User Experience: A Practical Guide to Research*. Rosenfeld Media.

Bucher, H.-J., & Schumacher, P. (2006). The relevance of attention for selecting news content. An eye-tracking study on attention patterns in the reception of print and online media. *Communications*, *31*(3), 347–368.

Buscher, G., Dengel, A., & van Elst, L. (2008). Eye Movements As Implicit Relevance Feedback. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems* (pp. 2991–2996). New York, NY, USA: ACM. https://doi.org/10.1145/1358628.1358796

Claypool, M., Brown, D., Le, P., & Waseda, M. (2001). Inferring user interest. *IEEE Internet Computing*, *5*(6), 32–39. https://doi.org/10.1109/4236.968829

Cool, C., Park, S., Belkin, N. J., Koenemann, J., & Ng, K. B. (1996). Information seeking behavior in new searching environment. In *CoLIS* (Vol. 2, pp. 403–416).

Cutrell, E., & Guan, Z. (2007). What are you looking for?: an eye-tracking study of information usage in web search. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 407–416). ACM. Retrieved from http://dl.acm.org/citation.cfm?id=1240690

Goldberg, J. H., & Kotval, X. P. (1999). Computer interface evaluation using eye movements: methods and constructs. *International Journal of Industrial Ergonomics*, *24*(6), 631–645.

Granka, L. A., Joachims, T., & Gay, G. (2004). Eye-tracking analysis of user behavior in WWW search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 478–479). ACM. Retrieved from http://dl.acm.org/citation.cfm?id=1009079

Harman, D. (2011). Information Retrieval Evaluation. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, *3*(2), 1–119. https://doi.org/10.2200/S00368ED1V01Y201105ICR019

Holmqvist, K., & Nyström, M. (Eds.). (2011). *Eye tracking: a comprehensive guide to methods and measures*. Oxford ; New York: Oxford University Press.

iMotions. (2016, January 12). What is eye tracking and how does it work? Retrieved April 27, 2017, from https://imotions.com/blog/eye-tracking-work/

Just, M. A., & Carpenter, P. A. (1976). Eye fixations and cognitive processes. *Cognitive Psychology*, *8*(4), 441–480.

Kauppi, J.-P., Kandemir, M., Saarinen, V.-M., Hirvenkari, L., Parkkonen, L., Klami, A., … Kaski, S. (2015). Towards brain-activity-controlled information retrieval: Decoding image relevance from MEG signals. *NeuroImage*, *112*, 288–298. https://doi.org/10.1016/j.neuroimage.2014.12.079

Kelly, D. (2009). Methods for Evaluating Interactive Information Retrieval Systems with Users. *Foundations and Trends® in Information Retrieval*, *3*(1–2), 1–224. https://doi.org/10.1561/1500000012

National Eye Institute (NEI). (n.d.). Diagram of the Eye. Retrieved April 26, 2017, from https://nei.nih.gov/health/eyediagram/

Okazolab. (2016). EventIDE Quick Guide. Okazolab Ltd. Retrieved from http://wiki.okazolab.com/GetFile.aspx?Page=MainPage&File=EventIDE%20Quick%20Guide%202016.pdf

Poole, A., & Ball, L. J. (2006). Eye tracking in HCI and usability research. *Encyclopedia of Human Computer Interaction*, *1*, 211–219.

Poole, A., Ball, L. J., & Phillips, P. (2005). In search of salience: A response-time and eye-movement analysis of bookmark recognition. In *People and computers XVIII—Design for life* (pp. 363–378). Springer. Retrieved from http://link.springer.com/chapter/10.1007/1-84628-062-1_23

Ruthven, I. (2008). Interactive information retrieval. *Annual Review of Information Science and Technology*, *42*(1), 43–91. https://doi.org/10.1002/aris.2008.1440420109

Schall, A., & Romano Bergstrom, J. (2014). 1 - Introduction to Eye Tracking. In *Eye Tracking in User Experience Design* (pp. 3–26). Boston: Morgan Kaufmann. https://doi.org/10.1016/B978-0-12-408138-3.00001-7