TESS: A tool for optically measuring digital reading interactions from screen recordings

By

Katherine Anne Brady

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

February 28, 2019

Nashville, Tennessee

Approved:

Douglas Fisher, Ph.D.

Gautam Biswas, Ph.D.

Maithilee Kunda, Ph.D.

Amanda Goodwin, Ph.D.

Vikash Singh, Ph.D.

Gayathri Narasimham, Ph.D.

ACKNOWLEDGEMENTS

I would like to thank Vanderbilt University for providing funding for this project through TIPs (Trans-Institutional Programs) and VIDL (Vanderbilt Institute for Digital Learning) grants. I also want to thank Amanda Goodwin and Daniel Levin for allowing me access to data sets that took them months to collect. The work described in this document would not have been possible without access to this data. My collaborators, Sun-Joo Cho, Gayathri Narasimham, Amanda Goodwin, Jorge Salas and Douglas Fisher have helped me understand the many fields of research that had to come together for this work. Sun-Joo Cho and Gayathri Narasimham helped analyze this data and explain the statistical models needed to understand it. Amanda Goodwin and her students Tess and Dan spent countless hours explaining educational research goals and processes. Jorge Salas has been an immense help for data wrangling and cleaning. My advisor, Douglas Fisher, helped develop these ideas and met with me many times to work on communicating them.

The fourth chapter of this document is filled with tests of TESS's accuracy and usefulness that required hours of time from many volunteers. I want to thank all of these volunteers, particularly those that took the time to participate in more than one evaluation test or helped recruit other evaluators. Many evaluators gave detailed and insightful feedback that changed my perceptions of TESS's potential customers and how TESS-human teaming should work. I also got a lot of valuable insights from the eye-tracking working group that met monthly on Vanderbilt's campus. In this working group I learned how Daniel Levin's and Gautam Biswas's groups used eye trackers in a variety of digital environments.

Finally, I would like to thank the friends and family members who have helped me throughout this process. My mother has lent her expertise in visualization to helping me formulate and explain visualizations of TESS's output. My friend, Ajay Dev Chhokra, has talked through many image processing strategies with me and pushed me to cover more edge cases. Many of my friends and family members have acted as a sounding board for my ideas and frustrations over the years. I am deeply indebted to you all.

TABLE OF CONTENTS

Page
ACKNOWLEDGEMENTS . . . ii
GLOSSARY OF TERMS . . . v
LIST OF TABLES . . . vi
LIST OF FIGURES . . . vii
Chapter
1 Introduction . . . 1
2 Related Research . . . 5
2.1 Functional Specification . . . 5
2.1.1 Digital Reading . . . 7
2.1.2 Navigation . . . 10
2.1.3 Dwell Time . . . 13
2.1.4 Highlighting . . . 16
2.1.5 Time Series Data . . . 18
2.1.6 Tracking Work . . . 22
2.1.7 Privacy . . . 24
2.1.8 Functional Specification Summary . . . 25
2.2 Similar Tools and Methods . . . 26
2.2.1 Human Coding . . . 27
2.2.2 Integrated Logging Software . . . 29
2.2.3 Optical Logging Software . . . 33
2.2.4 Similar Tools and Methods Summary . . . 35
2.3 Related Research Summary . . . 36
3 System Design . . . 37
3.1 Finding Transitions . . . 38
3.2 Finding Text . . . 42
3.2.1 Matching Frames to Documents . . . 43
3.2.2 Line Matching . . . 45
3.2.3 Word Matching . . . 47
3.3 Word Color Detection . . . 48
3.4 Action Coding . . . 49
3.4.1 Scrolling and Zooming . . . 49
3.4.2 Highlight Coding . . . 52
3.5 Outputs . . . 53
3.5.1 Corpus of hOCR files . . . 54
3.5.2 Aggregated Action Files . . . 55
3.5.3 Visualizations . . . 56
3.6 Human-TESS Teaming . . . 58
3.7 System Design Summary . . . 60
4 Evaluating TESS for Correctness . . . 62
4.1 Finding Transitions . . . 62

4.2 Preparing Images . . . 65
4.3 Document Matching Tests . . . 67
4.4 Line and Word Correction . . . 70
4.5 Color Detection Tests . . . 72
4.5.1 Single Highlight Color Test . . . 73
4.5.2 Large Color Range Test . . . 74
4.5.3 Four Color Test . . . 80
4.5.4 Color Detection Test Summary . . . 82
4.6 Action Coding . . . 83
4.6.1 Scrolling . . . 84
4.6.2 Highlighting . . . 86
4.6.3 Action Coding Summary . . . 87
4.7 Visualization Tests . . . 88
4.8 Running Time Tests . . . 94
4.9 Evaluation Summary . . . 96
5 Reading Experiments that use TESS . . . 98
5.1 Scrolling . . . 99
5.1.1 Background . . . 100
5.1.2 Method . . . 103
5.1.3 Results . . . 106
5.1.4 Discussion . . . 107
5.2 Zooming . . . 110
5.2.1 Background . . . 110
5.2.2 Method . . . 111
5.2.3 Results . . . 113
5.2.4 Discussion . . . 115
5.3 Experiments Summary . . . 116
6 Discussion . . . 119
6.1 Accuracy . . . 120
6.2 Usefulness . . . 124
6.3 Discussion Summary . . . 126
7 Concluding Remarks and Future Plans . . . 128
Appendix
A Middle School Reading Dataset . . . 134
BIBLIOGRAPHY . . . 138

GLOSSARY OF TERMS

CSV (Comma Separated Values) a file format for storing spreadsheets that many programming languages can parse

DOM (Document Object Model) an interface that models HTML documents as trees

GUI (Graphical User Interface) an interface for interacting with a program using graphical commands (such as button clicks and menu selections) as opposed to typed commands.

hOCR an open standard that extends HTML for storing OCR output developed in 2007 by Thomas Breuel [34]

HTML (Hypertext Markup Language) the extension of XML used to write websites

IDE (Integrative Development Environment) an application for writing and testing a computer program

JSON (JavaScript Object Notation) a file format for storing data objects that was originally developed for JavaScript but is now widely used. The JSON format is newer than XML and is often promoted as less bloated alternative.

NAEP (the National Assessment of Educational Progress) a test designed to measure what U.S. students know in various subjects. Also known as The Nations Report Card.

OCR (Optical Character Recognition) an set algorithms that extract text from images

STEM (Science Technology Engineering and Math) a term policy makers and educators use to group disciplines which rely heavily on mathematics and are (on average) valued highly by employers.

stop word words that are so common in a language their presence in a text snippet conveys very little information about that snippet. Examples for English include 'the' and 'it'.

TESS (Text-position Evaluation from a Screen-recorded Session) the tool proposed in this document

XML (Extensible Markup Language) a general markup language that has been extended for hundreds of purposes and is designed to be readable by both humans and machines.

LIST OF TABLES

Table Page

4.1 The word error rate for the four frames with and without prepossessing and before and after the text correction steps. 66
4.2 'Macro' actions coded by visualization evaluators 89

A.1 Pre- Test and Survey Questions 136
A.2 Post-Test Questions 137

LIST OF FIGURES

Figure Page

2.1 A visualization of how TESS's output can be paired with gaze data. The fixations in the gaze data are shown as purple dots and the word positions found by TESS are shown as red boxes. Using TESS, researchers can describe fixation locations in terms of their position within an article rather than their position on the screen. 20

3.1 An overview of TESS's process of extracting information from a screen recording. 38
3.2 The process by which times when the screen changed are found. 40
3.3 An example of a pair of images that look the same to a human but have different pixel values. The center image shows the pixels in the first image which differ by at least 20% of the color spectrum (50 out of 256 pixel values) in one of the three color channels from the pixels in the final image. The unshaded pixels all differ by at least this much. 41
3.4 The fraction of tokens that uniquely match one document, even if a character is changed based on length in a corpus of 80 English Wikipedia documents 45
3.5 This figure shows the number of documents in a Wikipedia corpus that have various probabilities of sampling a word unique to the document. The x-axis shows the probability a sampled word occurs exactly once in a document. The y-axis shows the number of documents in the corpus with that probability. 46
3.6 A frame in which the OCR engine failed to recognize text from the article because the background was too dark. 51
3.7 An example visualization of scrolling behavior. The image behind the graph is a 'stitched image' and provides context for the vertical position of the screen. Note that unless the user zooms in, they will see the full width of the 'stitched image' at all times. 57
3.8 An example visualization of highlighting behavior 58
3.9 A visualization of TESS's scrolling results which shows the user zooming out 4 minutes and 47 seconds into the session. The screen usually starts maximally zoomed out, so this zoom is likely to be an incorrect reading. . . 59
3.10 A screen shot of TESS's interactive editor. The word currently being edited is highlighted with a thick red box. 60

4.1 A screenshot of the evaluator's view during the transition finding test. . . . 63
4.2 The transitions found by TESS and the human evaluators in the video clip. The times at 64
4.3 One minus the ratio between the similarity score of frame and the correct document and the largest similarity score between that frame and an incorrect document. The ratios are subtracted from one so that the closest values will be near zero. 68

4.4 One minus the ratio between the similarity of an hOCR file with the correct document and most similar incorrect document under different similarity metrics. The ratios are subtracted from one so that the closest values will be near zero. . . . 69
4.5 The number of words human reviewers marked incorrect in each frame before and after correction by TESS. . . . 70
4.6 A reviewer marking text incorrectly deleted by TESS. . . . 72
4.7 A snippet of the website in which each word is randomly colored from one of 25 color pairs. . . . 74
4.8 The eight colors used for the text and background of each word. . . . 75
4.9 The probability that words colored using a given color pair will be grouped with other words by the OCR engine as a function of the average brightness difference between the two colors. . . . 77
4.10 How an image snippet of a red and orange-yellow word with a white border is partitioned by the clustering and thresholding approaches. . . . 78
4.11 The median color values for each cluster in a green and orange-yellow word and a green and light blue word. Tess's difference function categorized the orange-yellow in the top image as light blue and the green in the bottom image as orange-yellow. . . . 80
4.12 The error rate and type of error (color detection or labeling) for different colors. All colors refer to the background color. When the background was dark blue the text was white. . . . 81
4.13 The average pixels scrolled plotted against the total number of downward scrolls. . . . 85
4.14 Visualizations of scrolling from different points in the distribution. . . . 85
4.15 The amount of time that words were marked the wrong color across three videos ranging from 2.5 to 4.5 minutes in length each displaying 400 words or more . . . 88
4.16 The visualization view for video clip 2. . . . 90
4.17 The self assessed understanding level of each participant after reading the instructions and watching the explanatory video. . . . 91
4.18 The absolute difference between the count for each action when the same evaluator coded the video and visualization of each clip. . . . 92
4.19 Participant counts of each action based on whether they were viewing visualizations of the clip or the screen recording video. . . . 93
4.20 The amount of time users spent on each video clip in each condition. . . . 94

5.1 Number of upward scrolls plotted against number of downward scrolls. . . . 108
5.2 The percentage of students who got questions related to the second section correct based on whether they zoomed in at least 10% while reading the first section. The differences are statistically significant for questions 11, 13, 14, and 15. . . . 114
5.3 Number of students who zoomed in various amounts while reading the first and second section of the article. . . . 115

Chapter 1

Introduction

Educational reading materials are increasingly available to students in digital environments [59, 126, 144, 193]. The use of digital education materials is likely to increase as economic trends continue to drive down the price of electronic books and the devices needed to access them [164, 172]. However, little is known of how electronic texts should be designed to increase learner comprehension [192, 193], engagement [143] and enjoyment [104, 115, 125, 149, 202, 211, 224]. Expanding our understanding of design principles for digital reading environments will require extensive study of these environments. Currently, studies of digital reading are hampered by the available tools for monitoring user engagement. This dissertation presents TESS (Text-position Evaluation from a Screen-recorded Session), a tool that tracks and codes user actions in reading environments using screen recordings.

TESS improves on previous tools by focusing on behaviors that are of interest to the reading research community, working with all stimuli, and scaling for studies with hundreds of participants. There are existing tools that fulfill some but not all of these requirements. Unfortunately, effectively studying reading behaviors in authentic settings cannot be done with partial solutions. The following chapters discuss the problem TESS was built to solve, existing tools and why they do not solve the problem, and experiments that show TESS works as it was designed to and how it has been successfully used for scientific studies.

TESS is designed to:

1. Be easy to setup with any stimuli

2. Track text movements and interactions

3. Scale for larger studies

Existing tools don't do these things. Tools that integrate with the software used by participants cannot be setup with arbitrary stimuli. Existing optical tools do not track text movements and interactions. Currently, many reading researchers use human coders to track participant actions, but this does not scale for larger studies. The strengths and weaknesses of existing tools are discussed in more detail in Chapter 2.2.

There is a lot of diversity in digital reading environments. Software for reading ranges from web browsers to PDF readers to video games to text editors. The hardware this software runs on also has a lot of variance from large desktop monitors to mobile devices and even wearable devices like smart watches, which currently allow users to read short news articles and may expand their reading options over time. TESS uses video recordings of participants' screens as its input. By not relying on other software packages, this ensures that it works with all current reading environments and those that haven't been invented yet. On most devices, video recordings can be captured using inbuilt screen recording software. When this is not the case, a camera can be setup to record the screen.

TESS brakes the video input into segments based on when the screen changed. The color and location of each word on the screen in each segment are extracted by TESS using OCR (Optical Character Recognition) and pixel grouping. It uses text files which describe the readings used in the study and colors input by the researcher to correct mistakes in the extraction process. Once information on word positions and colors is extracted from each segment, TESS aggregates information across segments into 'micro' actions like scrolling and highlighting or pairs the information with other data sources such as eye tracking information to measure which words participants looked at most often. These 'micro' actions can be further aggregated to create 'macro' actions like editing a highlight.

In this context 'micro' actions are actions that take less than a second to complete or can be described by the number of short time intervals they occur in. For example scrolling can be described by the number of times the text moves in a series of $\frac{1}{2}$ second or $\frac{1}{100}$

second time intervals. Using $\frac{1}{100}$ second intervals will give a more precise measurement of how often participants scroll, but both window sizes are small enough to be useful. A 10 second time interval is too large to capture whether a participant slowly scrolls from time to time or moves the page in short 'bursts' that only last a second. Significantly, 'micro' actions, while often hypothesized about as discussed in Section 2.1, are rarely studied since human coders find it difficult and tedious to log 'micro' actions, and existing computational tracking tools are incompatible with many digital reading environments.

Researchers who wish to study 'macro' actions that take more than a second can benefit from first coding 'micro' actions with a tool like TESS. In cases where the 'macro' action can be easily described in terms of 'micro' actions, aggregating TESS's output instead of coding the actions with human coders can save time for lab assistants. When 'macro' actions are more subjective, human coders may find it easier to identify actions from a visualization of 'micro' actions than from watching participants in lab or reviewing a screen recording. The helpfulness of an example visualization of 'micro' actions for coding 'macro' actions is discussed in Section 4.7.

Since the text files and colors for correcting mistakes in TESS's extraction step must be entered by the researcher, TESS is most useful when a large number of users are asked to read a small set of documents. This is not to say that the range of experiments that would benefit from a tool like TESS is small. Even in experiments where participants are allowed to edit a lot of the text, TESS can be used to track behaviors like window switching (which can be monitored by tracking static text snippets like window headers) and errors encountered (the text of possible error messages is often known to researchers). The need for research into how to design good digital reading environments is well documented and growing as explained in Section 2.1. Publishers of print media are able to draw on centuries of reader observations to create the best artifacts for a range of audiences. Despite the fact that relatively little is known about how to make a good digital reading artifacts, electronic books are already used in classrooms. TESS helps educators and publishers by making

it easier to study how readers use digital texts and which behaviors have the strongest correlation with learning outcomes.

Compared to traditional paper reading environments, relatively little is known about how digital reading environments affect comprehension and enjoyment [104, 115, 125, 143, 192, 224], but the use of digital texts for education and pleasure is spreading rapidly. The quick adoption of new digital reading environments is likely to continue due to their relatively low cost [164, 172, 179], and the large amount of venture capital currently invested in educational technology start-ups [37]. As a result, it is vital that researchers investigate how readers experience digital reading environments and offer actionable advice for content designers. This is a large research goal. Digital reading environments lack many of the physical constraints of paper based reading environments which presents content creators with a much larger design space. The speed with which research findings can be acted upon is also faster in digital reading environments. Corrections to paper books must wait on publication, distribution and buying cycles, but digital texts can be edited and pushed to readers' devices remotely [13, 111, 173]. They are also cheaper to distribute if readers already have a device to view them on and can be more portable [211]. Thus, findings from research on digital reading environments can be applied faster and with less expense than research on paper reading. TESS enables researchers to quickly study existing digital reading tools and provide recommendations to readers, schools and software designers.

Chapter 2

Related Research

This chapter discusses both the evidence that a tool like TESS is needed and explains why existing tools are insufficient. These discussions are divided into two sections. Section 2.1 goes through studies and research areas that would benefit from TESS. This section shows there is a demand for a tool like TESS and discusses some of the ways TESS could be applied to answer existing research questions. Section 2.2 covers existing tools that solve some of the problems discussed in Section 2.1 and how TESS improves on these solutions.

2.1 Functional Specification

How students read both in and out of school is changing drastically [126]. As electronic reading devices such as laptops, tablets, and smart phones become cheaper, the fraction of educational reading material that students consume from paper is shrinking. Research by psychologists and educational specialists have uncovered differences in how students learn from paper and digital reading formats [143, 228], but there is less research into what causes these differences and how design decisions in digital devices affect learning. One reason for this gap is that researchers don't have access to tools for studying micro actions inside closed digital reading applications. 'Closed' applications do not publish their source code or provide interfaces for tracking software to monitor the application. Many of the software packages currently used in schools are 'closed'. Most studies that investigate 'micro' reading behaviors do so using 'open' software specifically designed for the study [5, 42, 150, 213]. TESS uses screen recordings to study 'micro' actions in any digital reading environment.

TESS extracts the location of words in a video at each time point using OCR that has been corrected using the text of stimuli files. Once the text locations are known, they can be used to compute reading actions such as scrolling and zooming. Highlighting can also be tracked by categorizing the pixel colors at a word's location. TESS will help researchers studying electronic book design by uncovering how different readers interact with the text. It will help educators by showing them which reading strategies are used by the strongest students (these strategies can be taught to other students). It can help individuals by showing them their own reading behavior in a variety of settings. Unlike existing systems, TESS can be applied to any electronic reading environment, allowing researchers to study the tools currently used in schools.

Tools for studying digital reading must

1. Be easy to setup with any stimuli
2. Track text movements and interactions
3. Scale for larger studies

If a tool only works with some stimuli, then researchers who wish to study the reading environments currently used in schools (over which they have no control) will not be able to use the tool. Even if these researchers partner with software designers to prototype new reading environments based on their research, they will not be familiar with tracking tools that they have not previously used. Thus, even in contexts where an integrated tool could be helpful, it is less likely to be used. A tool that cannot track text movements and interactions would miss most of the data in a reading study. Tools that track other things, such as mouse movements and button clicks, have been successfully used to study behavior during some reading tasks [9, 72, 89]. However, these actions are rare in many reading environments, where the most common actions are scrolling or page turning. Tracking tools that do not scale to larger studies do not provide enough value for researchers to justify the overhead of setting them up and learning how to use them. All tools take time to understand, in order

to be adopted by an overworked group like the research community, a tool must offer a big payoff for learning how to use it. Small data sets can be closely analyzed by research assistants, so any tool that only works for small studies is unlikely to provide much more value than hiring extra staff.

2.1.1 Digital Reading

How we read is constantly evolving. Until the first century A.D., Europeans read from scrolls rather than books [49]. The advantage of books with pages over physical scrolls is that they can be read out of order more easily than a scroll, and there is evidence that early Christians often used this technology to read the Bible out of order [197]. More recently, the invention of putting spaces between words and standardizing spelling made it easier to read text without sounding it out and disturbing anyone nearby [184]. The latest disruption is digital reading environments, which allow readers to view even scrolls out of order via hyperlinks.

In any reading environment, there is a large space of design decisions. For example, when printing a book a publisher must decide how thick the paper will be, how wide to make the margins for note taking, and whether to have cutouts for quickly finding certain locations in the book (such as the start of each letter in a dictionary). For paper books, publishers can refer to centuries of book usage to make these decisions, but digital reading artifacts have only been widely available for a few decades. To fill this gap in knowledge and ensure that digital textbooks are meeting the needs of learners, researchers must study a range of questions including: how do readers use digital reading materials, how does the design of digital reading materials affect learner goals, and what design decisions annoy readers.

The most common method for investigating this final question is to ask students about their preferences between various reading environments. Many studies have been done on preferences between digital and paper textbooks among university students. While both

universities [172] and students [164] report opting for electronic textbooks to save money, most students report that they prefer studying from paper books. The reasons for preferring paper books range from easier note taking in paper books [125, 202, 224], better focus [104, 211, 224], less eye exhaustion [125], and easier between-section navigation [149]. Less research has been done on preferences between different types of digital reading devices, but a 2011 study by Mitchell Weisberg found that students prefer reading from tablets to reading from laptops [211]. Weisberg also found that once the software on the tablets he gave students was updated to make note taking easier, many students reported preferring tablet reading to paper reading. The goal of this dissertation is not to replace paper reading with digital reading, but rather to facilitate the study of digital reading so that the design decisions digital content creators make are as informed as those made by paper publishers.

While reader preferences are important, how much learners are gaining from using various reading environments also matters. Like preference studies, investigations of comprehension in digital reading environments have primarily concentrated on comparing digital reading to paper reading. Recent meta-analysis of this work has found that while effect sizes are not large, most studies with significant results have found that students learn better from paper texts than digital texts [59, 193]. This result is not unexpected given the longer history of developing paper reading materials for classrooms, but Delgado et al. also found that the inferiority of digital reading environments has not decreased over the last two decades [59]. Thus, better electronic textbook design will not naturally emerge without research into how modern electronic texts are failing students. TESS provides a method for closely examining what behaviors are linked to decreased comprehension and using these findings to quickly iterate on existing designs.

There is evidence that the effect size of comprehension differences between digital and paper reading is affected by the subject of the text being read and the length of the stimuli. A study by Margolin et al. that tested subjects on narrative articles rather than textbooks found no difference between paper and digital reading [145]. A 2014 study by Mangen and

Kuiken found that readers using an iPad were less transported by a story than those reading from paper when they were told the article was non-fiction [143]. However, readers who were told that the same text was a fictional story experienced no difference between digital and paper reading. In 2017, Singer and Alexander found that the effect between digital and paper on collecting key points was larger for a book excerpt than a newspaper article, both because the digital readers did slightly better on the newspaper article and because paper readers performed worse on the newspaper article [192].

There are many potential reasons that digital reading may be better for some subjects than others. Readers who are used to reading novels linearly and don't like to mark up their books may not miss the navigational and note taking options available in paper environments when they read novels digitally. However these tools are often viewed as essential when going through examples in a textbook. Most readers who are currently adults did not study from digital textbooks when they were in school, but currently get a lot of their news from digital newspapers. Thus these readers may simply feel more comfortable studying from paper books and reading news from digital devices. This hypothesis has been dubbed the 'shallowing hypothesis' based on the idea that digital devices are primarily marketed and used for 'shallower' reading tasks. Researching how reading environment design affects readers across a range of subjects requires many studies of the same behaviors with different stimuli. TESS is well suited to assist these studies since it can easily be set up with any stimuli.

A few hypotheses have been put forward to explain the differences in comprehension between paper and digital reading. One is that the navigation methods used in paper environments are better for comprehension than those used in digital reading environments. This hypothesis is discussed further in Section 2.1.2. Another explanation has been named the 'shallowing hypothesis' and explains the difference by claiming that readers find it difficult to concentrate when reading from digital devices because they associate digital devices with distracting applications like social media.

The ‘shallowing hypothesis’ states that exposure to media devices decreases reflective thought [6]. This hypothesis has been widely championed by Nicholas Carr in an article for the Atlantic and a subsequent book [44, 45]. While there is some evidence that frequent use of social media and instant messaging decreases reflective thought [6, 140, 201], there has been less research into whether all digital reading negatively impacts reflection. A 2010 survey of experienced readers found that distracting reading environments are not limited to social media sites. Readers reported that most articles they read online had distracting adds embedded in the text [90]. In order to focus on the article, the readers said they tried to scroll the adds off the screen. A study in 2016 compared reading in an environment with ads to one in which all ads and pictures have been removed. They found that readers performed better in the environment without ads [70] but didn’t collect data on how readers acted differently in the two environments. TESS could be used to measure expert and novice behavior on websites with and without ads. These measurements would allow researchers to measure the difference in screen time ads get between expert and novice readers. They could also be used to measure whether expert readers change their behavior when the digital reading environment is designed to help readers focus.

2.1.2 Navigation

Interest in how readers navigate digital reading environments comes from several sources. Some reading researchers believe observed differences between digital and paper reading environments can be explained by how readers navigate in each environment [193]. Other reading researchers have suggested that better navigation tools in digital environments may improve reader enjoyment and comprehension [69, 132]. Researchers studying knowledge retrieval for libraries or search engines have found that some navigational tools help users find information faster [11, 55]. TESS can help researchers characterize navigation within and between documents by tracking which documents a participant is looking at and their position within the document.

In 2017, a survey of 36 studies comparing digital and paper reading comprehension found that the studies reported no difference or a slightly better outcome in the digital environment when the stimuli text contained 500 words or less and better outcomes in the paper condition when the stimuli contained 500 words or more [193]. Based on these results, the authors of the survey hypothesize that methods of navigating documents that are too large to be displayed on a single screen, like scrolling, are responsible for poor reading comprehension in the studies with longer stimuli.

The hypothesis that scrolling negatively affects reading predates this 2017 review. In 1997, O'Hara and Sellen asked members of their lab to read a scientific article either on a desktop or on paper and summarize it [156]. Subjects reported that the task was easier on paper because scrolling was difficult in the digital version and note taking tools were better in the paper version. Ten years later, Erik Wstlund asked students to collect information on a desktop in which text was navigated either using a scroll bar or page links. He found no difference in test results between the two versions, but participants who had to use the scroll bar said it was more mentally taxing [208]. In 2008, Kim and Huynh found that while there were no significant differences in test results between students who took the test on a computer and those who took it on paper, the test takers using the computer did worse during the reading portion, which required them to read a longer article than any other part of the test [107]. A 2011 study of students reading articles on an iPad compared reading comprehension between students using paging and scrolling to navigate the articles. The authors found that participants using the scrolling environment did worse on narrative texts, but there was no difference for procedural articles [73]. Characterizing how students scroll with a tool like TESS would allow researchers to test whether the observed effects of scrolling environments are linked to certain scrolling behaviors or impact all readers equally. An example of a study like this is discussed in Section 5.1.

Digital environments present a much larger design space for reading navigation than analog environments because they have fewer physical constraints. Some researchers have

looked beyond paging and scrolling to explore how alternative navigation methods may help digital readers. In 2006, Leigle and Janicki asked 63 undergraduates to read a series of lessons [136]. Each lesson had hyperlinks embedded in the lesson text and as well as previous and next buttons at the bottom of the page. The table of contents was shown at the side of the page, and readers could click on it to access the beginning of each module. Leigle and Janicki found that whether learners preferred to follow hyperlinks embedded in the lesson or click the next button at the bottom of the page (and go through the lessons in order) correlated with how learners were categorized on Kolb's Learning Styles and Learning Preferences scale. Those learners who used the navigation strategy that correlated with their learning style category did better on a post test than those who did not.

Cockburn et al. developed a reading environment in which readers can view an entire book as a grid of thumbnails displaying the contents of each page [55]. To read a page users can click on its thumbnail. The page that a reader was most recently on is highlighted in the thumbnail view so readers don't loose their place. Cockburn et al. compared the speed of user navigation between their environment, a scrolling environment and scrolling environments enhanced with clickable thumbnails. They found that their 'page thumbnail' environment enabled users to navigate faster.

Atterer and Lorenzi build an environment in 2008 that color codes the scroll bar in a scrolling environment to show users which parts of a webpage receive the longest dwell times [11]. A user study comparing this system to the existing table of contents navigation system for Wikipedia articles found that all participants preferred the color coded scroll bar. On average, participants performed 7 out of 8 tasks faster when using the color coded scroll bar, but only a few of these differences were significant.

In 2013, Li et al. asked 60 undergraduates to read a passage, take a post test and then come back a week later and take another post test after reviewing the stimuli [132]. The authors built a reading environment for the study that inserted a clickable toolbar at the bottom of the text which was designed to encourage users to take surveying and question-

ing strategies while reading. The authors hoped these strategies would help readers build a contextual map of the text. Each participant was assigned to one of 3 reading conditions: using a scrolling environment without guidance, using a scrolling environment and receiving guidance on surveying and questioning strategies, and using the environment with the clickable toolbar. Students who were given the toolbar completed navigational tasks faster and got better scores when they came back for the final test.

In 2015, Franze et al. developed a reading environment that opened links to other parts of the document in a secondary 'split view' window so the source of the link remained on the screen [69]. They compared this environment to one that used a single view and only showed the targets of links. The results showed no difference in reading comprehension between the two environments, but participants said they preferred the split view.

In the above experiments, a new reading environment was built for each study. Thus, tools that track environments by interfacing with the software will only work with these environments if the researchers building the environment ensure that their environment matches the interface required by the tool. Most researchers are on tight schedules and a lot of environment building is done by graduate students without much software engineering experience. This makes it difficult to use interfacing tools with environment prototypes. TESS solves this problem by working from what is displayed on the screen rather than the underlying program. This means it should work as well in new reading environments as in existing ones.

2.1.3 Dwell Time

How long a document or section is displayed on the screen during a reading experiment has been linked to user interest in knowledge retrieval studies. In 2001, Claypool et al. found that the amount of time web searchers spent on a webpage had a strong correlation to the relevance of the page as rated by a computational recommender system [54]. In 2014, Balakrishnan and Zhang found that dwell time was as precise an indicator of document

relevancy as other passive feedback measures like text selection and whether or not the user reviewed the content [17]. Models that integrated multiple passive feedback channels outperformed models that only looked at one attribute or only relied on the content of the document.

Other studies have found that dwell time can be an indication of interest, but is not equally meaningful in all contexts and varies significantly between users. In 2001, Kelly and Belkin ran a study that found dwell time was not significantly related to participants' self-rating of document relevance [105]. Based on this, they suggest dwell time may be relevant in some knowledge retrieval scopes but not others. In 2004, Kelly and Belkin ran another study on a larger set of search tasks that lasted fourteen weeks [106]. This longer study only contained 7 participants. Some participants showed a relationship between display time and document relevance but others did not. On average there was no relationship.

In 2008, Liu et al. built a model of page relevance based on the links users followed and the time they spent on each page [139]. They found that this model outperformed historic models like PageRank. Like Kelly and Belkin, Liu et al. did not always find dwell time information helpful. To increase the accuracy of their model, Liu et al. treating dwell time like a noisy variable. If the dwell time for a page exceeded 30 minutes, it was assumed that the user had left their computer and come back to it, so the dwell time observation was thrown out and replaced with another dwell time observation for that page. A study by Kim et al. in 2014 also found that modified dwell time measurements were useful for predicting user satisfaction [110]. Their model was able to predict the satisfaction users received from clicking on links from dwell time if they conditioned dwell time on the page's topic, length, and ease of reading.

When the screen size is small, it makes more sense to measure the dwell time for parts of the document instead of for the document as a whole, since less of the document can be shown on the screen at once. More knowledge retrieval researchers are measuring dwell time in this way as the use of small screens increases. In 2009, Buscher et al. found that

using paragraph level dwell time to re-rank search results and expand queries produced results that were as good as those based on eye tracking data [41]. In 2014, Yi et al. found that DOM (Document Object Model) item level dwell time could be used to predict user engagement with a webpage if the engagement model was trained separately for each user and normalized for the screen size [222].

In 2014, Lagun et al. measured the correlation between the dwell time and gaze duration for items in a web page's DOM when the webpage was viewed from a mobile phone [121]. They found that the phone's screen typically only displayed three to four DOM items at a time and there was a strong correlation between an item's dwell time and the total gaze duration recorded for the item with an eye tracker. This suggests dwell time can be used as a proxy for gaze at the paragraph level on mobile devices. Two years later, Lagun and Lalmas looked at the relationship between dwell time for different parts of news articles and user engagement [122]. They found that document level dwell time was not a good proxy for engagement since unengaged users may spend a lot of time staring at pictures at the top of the article while interested users may quickly skim the article. However, by modeling the time a participant spends on each part of the article as a topic model Lagun and Lalmas were able to predict user engagement.

Dwell time may be complicated to interpret across users, but there is evidence that users can easily extrapolate meaning from their own dwell time data. In 2008, Atterer and Lorenzi built a tool that shows website users which parts of a document they previously dwelt on the longest [11]. Participants were asked to revisit Wikipedia pages multiple times while completing a set of 8 tasks. Participants did better on the tasks when they were shown their dwell time data and described the experience of using the website with this data as more positive.

Dwell time has not had as much interest in the educational reading research community as it has enjoyed among knowledge retrieval researchers, but there is evidence that education researchers should study dwell time. In 2015, Sadallah et al. built a system for online

courses that measured how long students spent on each document in the course [183]. They asked course authors whether they valued this feedback and 71% said yes. The tools used to study dwell time in the experiments described above all require the stimuli to be a website, which is unrealistic for many educational studies, many of which study reading behaviors in PDFs and ebooks. TESS allows education researchers to measure dwell time in offline environments as well as on websites.

2.1.4 Highlighting

Active engagement with learning resources has been shown to result in better outcomes than passive engagement [23, 114]. Texts can encourage active learning by asking readers to solve questions, but the most common method of active engagement inside reading environments are annotations. Digital reading environments usually don't have as many annotation tools as paper environments [147, 211]. There is evidence that the highlighting and commenting tools currently available in many digital reading environments are not as conducive to learning as using a pen with paper [23].

The reason why digital annotation tools are not as helpful as paper annotation tools is still an open research question. One hypothesis, advanced by Mueller and Oppenheimer, is that students learn more from note taking if the act of recording a note takes longer [154]. Mueller and Oppenheimer developed this hypothesis after conducting a study in 2014 that found that students did better on post-tests when they took notes by hand than when they typed their notes (the students typed faster than they wrote by hand). An examination of the notes taken showed that typed notes usually resembled a transcript of the lecture video while hand written notes contained less text and synthesized the lecture more. Similar studies have found that comprehension differences between typing and writing notes longhand depend on the nature of the post test. In 2011, Bohay et al. found that there was no difference between typed and hand written notes for performance on a multiple choice test [30]. A study by Bui et al. in 2012 asked students to type notes during a lecture [36]. Half the

students were asked to transcribe the lecture and half were asked to take organized notes. The students who transcribed the lecture did better on a test administered immediately after the lecture, but those who organized their notes did better on a test administered 24 hours after the lecture.

A thorough examination of Mueller and Oppenheimer's hypothesis would require examining the role of time in annotation tasks outside of lecture notes, such as highlighting. TESS records highlighting and text selection in most colors and pinpoints when highlights are made, edited and deleted. This information could be used to determine whether students who take longer to make or edit highlights experience greater learning gains than those who highlight quickly.

Highlights and text selections can also be studied as potential indications of interest and summary tools. In 2012, White and Buscher found that text selections could be interpreted as a passive indication of interest to improve search results [213]. A study by Balakrishnan and Zhang in 2014 found that text selections are a more precise indication of interest than other passive measures such as reviewing the page [17]. In knowledge retrieval tasks, text selections are relatively rare [42], so search engine researchers have not investigated text selections as thoroughly as attributes like dwell time. However, in educational reading contexts where highlights can be saved for later review, highlighting is more common. TESS's highlight tracking can be used by researchers looking at a variety of stimuli who wish to study user interest.

There is evidence that students enjoy using highlighting tools while they read. A 2011 study by Weisberg asked business students to describe their preferences between paper books and electronic books on tablets after using both books [211]. In the first two semesters (when the tablets did not have highlighting capabilities) the students preferred the paper books, but once the tablets added highlighting tools in the third semester a plurality of the students preferred the tablets.

Digital reading environments may currently have inferior annotation tools, but there is

a lot of potential for building improved tools. Digital highlights and notes can be extracted from the source material and shared easier than paper annotations [102]. TESS makes it possible to extract text selections from a screen recording for later review even when the reading environment does not provide a method of saving highlights. In 2005, Chi et al. prototyped a tool that automatically highlights parts of a document based on a user's search to make skimming for information easier [52]. By using TESS and an eye tracker, researchers can observe the correlation between the positions of automatically highlighted words and gaze to test the value of automatic highlights for guiding reading.

2.1.5 Time Series Data

There are several tools that track user actions over time such as mouse trackers, keystroke loggers, eye trackers, brain scanners and heart rate monitors. These tools are increasingly being used by researchers as they become cheaper, more accurate, and easier to setup [116, 186]. Unfortunately, most tools in this category record the time that a participant moves the mouse or experiences an elevated heart rate as a time series, but not what was displayed on the screen at that time. This makes the time series information less useful for answering research questions. TESS bridges this gap for digital reading studies by monitoring what is displayed on the screen throughout the study so the time series information can be linked to the stimuli.

The mouse and keyboard are controlled by a device's operating system, so they can be tracked independently of any software displaying the stimuli. However, in order to measure mouse position relative to the content of the stimuli, the content being displayed must also be recorded. TESS's output can be paired with the output of a mouse movement or keystroke logger by matching timestamps.

There are already tools that record documents displayed on the screen with mouse movements and keystroke logs for some stimuli types. In 2001, Mueller and Lockerd conducted a user study on the value of mouse movement data which was recorded by a

modified webpage [153]. They found this data helpful for modeling user interest in search tasks. In 2006, Atterer et al. made a JavaScript plugin to track mouse hovers and clicks in unmodified webpages [12]. As a case study for their tool, Atterer et al. asked 12 participants to find edit instructions on a Wikipedia page and create a calendar appointment on a web calendar. From the information gathered by their tool they were able to determine that only 4 of the participants took the optimal path through links on Wikipedia and document mistakes users made while making the calendar appointment. Based on these results, Atterer et al. suggest that mouse data is useful for studying webpage usability. In 2008, Guo and Agichtein used mouse movements (tracked by a JavaScript plugin for Firefox) to distinguish between navigational and informational searches [78]. They note that mouse clicks and hovers are insufficient for categorizing these types of searchers, proving that mouse movements contain value beyond click through information. Since 2008, several researchers have used mouse movements on websites to study search strategies [10, 42, 80, 93, 120]. TESS allows studies of mouse movements during reading to be conducted outside of websites.

Tools that log the state of word processing software in addition to keystrokes have been used to study writing processes. In 2006, Leijten and Waes built a keystroke logger that records keystrokes and the content of the screen for common word processors on Macintosh computers [128]. In 2016, Torrance et al. used keystroke logs, mouse tracking and eye tracking to measure how much time writers spend re-reading their work [200]. In 2017, Chan logged writer's keystrokes using Leijten and Waes's tool, and interviewed each writer about their cognitive process immediately after the study [47]. She found that actions recorded in the keystroke data such as pausing and revisions were correlated to how the writer described their cognitive state at the time. TESS is developed for reading rather than writing experiments, since the text used in the stimuli is a required input to correct the OCR. Keystroke logs that describe notes taken by the user (which TESS ignores) or key-presses used for navigation such as 'page up' and 'page down' can be used to augment TESS's

output. In future work, the text used to correct the OCR can be modified by keystroke information to enable TESS's use in writing studies.

Even when not connected to the stimuli, mouse and keystroke information has been used to draw conclusions on user behavior. Both mouse movements [174] and keystroke logs [129] have been shown to contain enough information to uniquely identify users. In 2009, Wengelin et al. used keystroke and eye tracking data to study the cognitive processes during text production [212]. TESS would allow these researchers who are not currently served by existing tools to connect mouse and keystroke information to their stimuli.

Mouse movements have been studied by several researchers as a proxy for gaze [50, 79, 92, 93, 135, 180]. How well this works appears to depend on the task participants are performing, but there is evidence that some researchers could use mouse tracking with TESS to approximate where readers' eyes are fixating if they don't have access to an eye tracker.

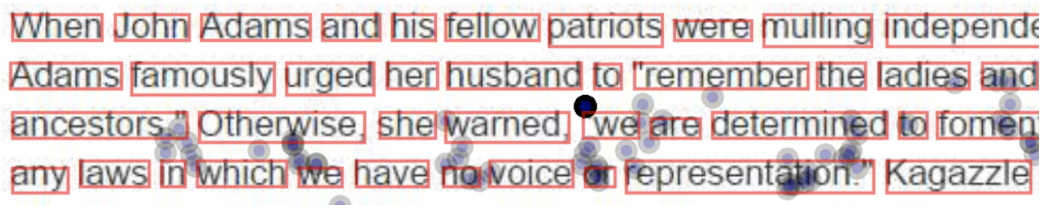


Figure 2.1: A visualization of how TESS's output can be paired with gaze data. The fixations in the gaze data are shown as purple dots and the word positions found by TESS are shown as red boxes. Using TESS, researchers can describe fixation locations in terms of their position within an article rather than their position on the screen.

As eye trackers improve, they open a new avenue for studying reading. Researchers have used eye trackers to monitor how often users move their eyes in various environments [228], which parts of a computer screen users fixate on the most often [38, 39], and count the fixations of readers at various proficiency levels [119]. Unfortunately, modern eye tracking software can tell researchers the Cartesian coordinates of where a participant is gazing on a screen, but only records what is displayed on that screen as a video. This works if the stimuli is a still image or a video, but does not connect gaze to the stimuli when the

participant has control of the screen. When participants use normal reading environments and have control over when text is scrolled or a page is turned, figuring out what is displayed under a participant's gaze at each point in this video recording is non-trivial. There has been some work on estimating what users read without connecting their gaze to what they read. For example, Kunze et al. found that they could estimate the number of words users view from the number of fixations in their gaze data [117]. More recently, Bhattacharya and Gwizdka used the speed of participant's fixations to determine whether they were closely reading a section or skimming it [27]. TESS eliminates the need for this by providing the Cartesian coordinates of all words on the screen, so gaze fixations can be mapped to the stimuli not areas of the screen. A visualization of how fixation coordinates and TESS's output can be combined is shown in Figure 2.1.

Several methods have been developed for measuring emotion during reading experiments. In 2010, Feild et al. used pressure sensitive chairs and mouse buttons to predict frustration during web search [66]. When they combined this data with software that detects emotion from a video recording of the participants face they were able to model frustration with 66% accuracy. Feild et al. also tracked user's search query behavior using a plugin for Firefox that told them how long users took to perform a task, how many pages users visited, and what queries users typed. These query statistics were more accurate for modeling frustration than the emotional response sensors. TESS would allow researchers to track dwell time and articles visited regardless of whether the stimuli was shown using Firefox or not. Researchers could pair this information with emotional response data using timestamps.

In more recent years, emotional response sensors have improved. In 2016, Fabian Benitez-Quiroz developed software to recognize emotions from videos of users' faces [65]. They were able to make their software more robust by training it on a diverse sample of faces that included a wide range of skin colors and ages. As emotional response sensors become more reliable and easier to use, more researchers will want to use them. TESS allows

these researchers to pair any emotional response data they collect with precise information about what participants are viewing.

2.1.6 Tracking Work

Digital reading environments are not limited to electronic books and websites. Researchers are also interested in tracking user actions in IDEs (Integrative Development Environments) [46, 131, 142, 159, 163, 170, 171], application switching behavior [99, 146, 158], notification management [43, 99], and menu selections [135, 182]. If the text that participants are reading is not known in advance, such as when researchers want to track what participants are typing, then TESS cannot be used since the OCR detected labels are insufficiently accurate. However, if researchers are interested in text that is not being edited, such as window titles, menu items, notification headers, or error messages, TESS can be used to track participant actions. The prevalence of text snippets which are not edited, such as button labels, is larger even in text editors like Microsoft Word. Tracking user actions related to these snippets can provide insights for improving worker productivity [21, 43, 99, 182] and educational materials [46, 131].

In 2005, Mark et al. observed workers in an information technology office as they went about their day [146]. The researchers found that workers frequently received interruptions throughout the day, but these only slowed work down if they forced the worker to change the 'sphere' of their work (i.e. forced them to think about a different project). Mark et al. suggest that work fragmentation should be measured according to how often workers must change 'spheres' not how often they change applications. One way of doing this is to categorize documents that workers are looking at and measure how often the main document on their screen changes categories using TESS.

In 2007, Iqbal and Horvitz found that they could get more precise information than Mark et al. on how interruptions were caused and how long they lasted by using a tracking tool [99]. The tool Iqbal and Horvitz used only works on Windows computers, but it

demonstrated the value of tracking window switching and alerts for categorizing disruptions. Iqbal and Horvitz were able to determine that workers lost 10 minutes per day to reading alerts and task switching. In 27% of cases, the main task was suspended for at least 2 hours. By tracking window switching and alerts across many devices using TESS, researchers could examine whether the alert management tools on Macintosh computers or tablets help workers stay focused.

Tracking and visualizing tasks can be used to help workers return to the mental state they were in before an interruption. In 2009, Cangiano and Hollan found that workers in a law office were better able to restore their mental state from the previous day if they reviewed a sped up screen recording of their work from the end of the previous day [43]. In 2014, Shreve et al. found that translators were better at catching errors made by another translator when they reviewed a screen recording of the other person translating the document than when they reviewed the other person's notes (which were made while making the initial translation) [191]. In 2015, Adam Rule suggested that using visualizations might help workers recall their previous mental state faster [182]. TESS collects user actions from screen recordings which can be visualized to highlight the most important actions for restoring context. Some example visualizations are shown in Section 3.5.3. Researchers using TESS may work with participants to develop visualizations that are better suited for work resumption.

Educators are interested in tracking when students get frustrated so course materials can be developed to concentrate on the subjects that cause the most confusion. In 2010, Carter and Dewan implemented a tool that flagged times that programming students experienced difficulty based on IDE events [46]. In 2016, Fwa Hua Leong found that adding keystrokes and click stream information allowed researchers to more accurately track participant frustration [131]. Using TESS to track user actions visually could help researchers track student frustration more precisely.

There are instances when a screen recording is the only record of a developer's IDE

usage. YouTube is increasingly being used to document and teach programming skills [142]. Researchers are interested in extracting the most relevant parts of these videos. Extracting these clips would allow educators to create summaries for programming students who may feel overwhelmed by the number of tutorials on YouTube [169], and link video clips to relevant portions of paper tutorials or discussion forums [171]. It would also help researchers compare the content between video and paper programming tutorials [204]. TESS could be configured to track common errors or code snippets in tutorial videos.

2.1.7 Privacy

Privacy is a larger concern in education research than most other fields, since participants in education studies are likely to be children. For this reason, it is very important that tools used by education researchers do not expose participant information. Many of the tools used by information retrieval researchers to track user actions upload user information to servers on the internet [9, 123, 206]. In studies that are carried out over the internet (as many information retrieval studies are), this does not add an extra security risk. However, in a classroom or lab study there is no need to transfer participant information using an internet connection, and uploading information to a third party server increases the probability that participant confidentiality will be breached. Thus, offline tools (like TESS) are preferable when they are available.

There are a lot of open questions in educational reading, and answering these questions will require researchers to track many participant actions. Unfortunately, most fine grained tracking information can be used to uniquely identify users. In 2004, Pusara and Brodley found that computer users could be identified by their mouse movements [174]. It is very likely that other 'micro' actions like scrolling patterns can also be used to identify participants. In a 2006 study, Tang et al. found that many participants were aware of this, so recruiting participants requires researchers to develop trust around privacy [199].

Some researchers have investigated methods of anonymizing small tracked actions

while preserving the information necessary for research [96, 129, 157]. Unfortunately these methods require the anonymizer to know what information is likely to be necessary for research and for the pipeline prior to the anonymization step to be secure. Most researchers cannot guarantee the security of a pipeline that transmits raw data across an internet connection, and participants may not feel comfortable being tracked by tools that were developed by large tech companies. Offline tools like TESS give more control to the researcher.

The importance of privacy in reading studies will vary based on who and what is being studied. Studies that track users through a website or browser usually ask users to consent to being tracked when they navigate to the study site [12], or justify tracking users who may not be fully aware of the privacy risks by noting that many search giants implement this level of tracking on all visitors [3]. One goal of reading research is understanding how reading behaviors differ between age groups. As a consequence, subjects are often children and must be consented before the study by a parent or guardian. TESS does not send any data across an internet connection. Researchers can move screen recordings of subjects using USB sticks to secure computers and TESS can be run locally.

2.1.8 Functional Specification Summary

The goal of studying digital reading is not to replace paper reading, but to help writers and publishers of digital texts design content and environments that are easy to use, foster education and enjoyable for their users. Digital texts are already being used in classrooms and workplaces around the world, but the effects of design decisions in these texts are largely unknown. Digital texts have a lot of potential since they are cheaper to copy than paper texts and digital devices do not have the same constraints as mechanical ones, but knowing how to build a good digital reading environment will take more research.

TESS is a tool for studying digital reading that:

1. Is easy to setup with any stimuli

2. Tracks text movements and interactions

3. Scales for larger studies

The sections above demonstrate that there is a lot of demand for a tool like this among researchers. Tracking how readers manipulate text to avoid distractions would provide insights for the 'shallowing hypothesis' [70, 90]. Analyzing correlations between tracked text movements and test performance would be useful for investigating the hypothesis that scrolling negatively effects comprehension [193]. Tracking how long a word is displayed on the screen can tell researchers which parts of a document are most interesting or hardest to understand for readers [54, 183]. Finding the correlation between the amount of time a student takes to select text and save it as a highlight and their test performance would provide evidence for or against Mueller and Oppenheimer's note taking hypothesis [154]. Combining text position information with the output of eye trackers would make eye tracking information more useful [27, 119, 228]. Tracking text in menus, notifications and error messages can help software designers make better tools for workers in a variety of fields [99, 131]. To answering these questions efficiently, researchers must be able to setup studies quickly without worrying about the software used to the display the stimuli, and TESS allows them to do that.

2.2 Similar Tools and Methods

This section gives an overview of existing methods for tracking users in a digital environment and discusses why these existing methods are insufficient for digital reading research. There are three categories of methods for coding participant actions in a digital environment:

1. Track actions using human coders

2. Automatically track actions as they happen using software that interfaces with the stimuli software

3. Automatically track actions from screen recordings using image processing

Human coders watch participant actions either in person or from a video recording and categorize what they see for quantitative analysis. This works well for 'macro' actions in small studies, but it is difficult for human coders to notice 'micro' actions that take less than a second. Software that interfaces with stimuli software limits what software can be used for study stimuli. If a researcher wishes to change the stimuli from a website to a PDF while developing the study, they would usually need to find new interfacing software for tracking. Software that automatically tracks actions from screen recordings (like TESS) does not have this limitation. Unfortunately, prior to TESS tracking tools that use screen recordings were not designed for digital reading experiments.

As mentioned in Section 2.1, tools for studying digital reading must:

1. Be easy to setup with any stimuli
2. Track text movements and interactions
3. Scale for larger studies

Human coding does not scale well and is vulnerable to errors. Software that interfaces with the stimuli software is not easy to setup and does not work with all types of stimuli. TESS tracks actions from screen recordings using image processing. It is not the first tool to track actions using screen recordings, but it is the first to be designed for reading experiments. Existing tools have been designed to study interactions with GUI (Graphical User Interface)s [18, 48, 72] or to study interactions with IDEs [19, 20, 21]. As a consequence, the actions that the tools track are button pushes and typing, not text movements or highlighting.

2.2.1 Human Coding

Coding by hand is the oldest method of tracking user actions. Traditionally, coders have recorded participant actions by taking notes while a study is administered. In some

cases, this is still the only option because study subjects may be concerned that electronic recording tools will capture nondisclosure protected software [146], may find the idea that a computer is tracking their every move too distracting [199], or researchers may feel that sitting in the room with subjects and recording their actions is sufficient for the study [88]. However, As computers have gotten better processors and are better equipped to record the screen and display the stimuli at the same time, researchers using human coders have moved to recording the screen using screen capture software or an over the should camera. This allows coders to discuss and refine their codes after a study has been conducted and opens the door to a lot of exploratory research that is not possible with pre-fixed codes. TESS also uses screen recordings as its input, so in many cases reading researchers would not need to collect more information while running their experiment if they use TESS than if they were using human coders, and TESS can be used to analyze previously collected data.

After running a study, researchers typically review a few sample videos to develop a coding scheme and achieve consensus. Each video is then reviewed manually by a small set of researchers who record whether the agreed upon codes are present. In the past five years, researchers have gotten more coders to annotate publicly available screen recordings by crowd sourcing the coding process [57, 108, 127, 210]. However, this option remains out of reach for many researchers, since showing participant recordings to a crowd would violate participant privacy, as discussed in Section 2.1.7.

Which information can be recorded by human coders is limited by the types of actions that humans can consistently notice. Examples of things humans notice easily include, whether readers scroll past information ‘quickly’ or ‘slowly’ [56], whether programmers are making ‘slow progress’ or ‘having difficulty’ [46], whether library users employ ‘linear’ or ‘nonlinear’ search strategies [25], the time participants spent on assigned tasks [97], how often readers ‘length check’ a webpage by quickly scrolling to the bottom and back up again [70], and which bugs among a predefined set participants fixed [7]. Notably,

these are all ‘macro actions’ that happen across the span of at least a few seconds and have been grouped into discrete categories. This limits how precise and nuanced codes can be. Computational coding systems can assist human coders by recording ‘micro actions’ that happen in less than a second and are difficult for humans to notice. Computational systems can also describe ‘macro’ actions in terms of ‘micro’ actions by using continuous variables such as the number of pixels scrolled or the time in seconds that text remains selected.

A survey of reading studies between 1992 and 2017 found that half the studies had at least 58 students and 15% had 100 or more [193]. Hand coding over 50 students is very time intensive. One way to increase the speed and complexity of human coding is to alter screen recordings so that coders can get more information out of them and review each video faster. In 2009, Cangiano and Hollan proposed a tool for lawyers to review the work of the previous day by watching a screen recording at five times the regular speed [43]. In 2012, Leiva and Viv suggested augmenting screen recordings with traces of recent mouse movements so mouse actions would be easier for humans to notice [130]. In 2017, Park et al. found that programming students were able to understand programs written by their peers better when they were shown a video of the code’s history in which the scrub bar was annotated with points that the code changed [163]. TESS increases the speed and quality of human coding by visualizing ‘micro’ actions so human coders can extract macro actions from the visualizations without reviewing the screen recording, as discussed in Section 3.6. A test of these visualizations is presented in Section 4.7.

2.2.2 Integrated Logging Software

The fastest and most precise method of tracking user interactions on the screen is to use software that records interactions as they happen by recording application calls. Unfortunately, this option requires the researcher to have a software package that can interface with whichever program they are using for the stimuli and have permissions to install the package on all study computers. The most common method of getting this permission for large

scale studies is to serve both the stimuli and the tracking software through a website, since browsers allow web applications to run without going through all the steps required of a desktop application. However, this option has several drawbacks for education researchers. It stops them from researching some of the most popular reading environments such as ebooks and PDF readers, introduces security risks, and forces studies to take place in areas with strong internet connections (which rules out a lot of schools).

Information retrieval researchers have done a lot of work to build tools for tracking users through websites. Most of these tools are built to find correlations between user interest and various actions. Example actions include mouse movements [3, 10, 78, 79, 89, 93, 120, 153, 180, 225, 226, 227], mouse clicks [17, 42, 94, 105, 226, 227], keys pressed [42, 160, 225, 227], time spent on each part of a webpage [11, 17, 41, 54, 68, 94, 106, 110, 121, 123, 138, 151, 214, 222, 225], webpages visited [4, 17, 225], scrolling positions [11, 42, 94, 105, 207, 225], search queries [24, 67, 198], and text selection [17, 42, 74, 213, 225].

As mobile devices have become a more popular method of accessing search engines, researchers have adapted what they track. For example, in 2011, Agichtein et al. tracked zooming in addition to pages visited to measure user interest [4]. In 2013, Guo et al. tracked touches instead of mouse movements to predict web search relevance [81]. Multiple researchers have adapted how they track dwell time for mobile devices by only tracking the part of the page that was displayed on the screen rather than treating entire webpages as units [121, 123, 124]. These interactions are increasingly becoming important for research on laptops as well as touch screens become more common in laptops, and more users are interacting with larger displays the way they interact with mobile devices.

Eye trackers have also changed the focus of information retrieval studies. In the early 2000s, eye trackers were bulky and difficult to use [186], but technical improvements have caused many information retrieval researchers to add gaze data to their studies [3, 8, 27, 39, 109, 121, 133]. Good eye trackers are still expensive, and eye tracking cameras can

only be used in lab studies (as opposed to studies administered over the web). To capture gaze information without an eye tracking camera, some researchers have experimented with using mouse movements [50, 79, 92, 135, 180] as a proxy for gaze, or capturing gaze with a web-camera as Papoutsaki et al. attempted to do in 2016 [161].

Many of these actions are interesting to educational reading researchers, but the tools developed by information retrieval researchers are not easy to integrate into stimuli for educational reading research. Information retrieval researchers are usually interested in designing better websites, so their studies are confined to web users with strong internet connections. The median bandwidth speed for a school in internet in the United States is 524 Kbps [63]. The average American phone user enjoys a download speed of 27.3 Mbps (53 times faster) and an upload speed of 8.6 Mbpd (17 times faster) [196]. Thus reading researchers who wish to study students in schools need tracking tools that don't rely on strong internet.

Even reading researchers who wish to conduct studies on online learners or in labs with strong internet connections usually need to custom fit the web technologies they use to their stimuli or design their study to work with existing tools. In 2015, Sadallah et al. used web technologies to log how long readers in an online course spent on each document and how often they revisited previously read documents [183]. The logging system they employed only logged when users requested a webpage, so they used the next request as the 'end time' for reading each document based on the assumption that learners were not navigating to other websites in the meantime. The authors were not able to distinguish between tasks (e.g. reading, searching, commenting) using the tracking software, and lamented that their study could not be more precise. Morey et al. were able to use mouse tracking web tools in conjunction with an eye tracking camera to precisely track which word participants were looking at in a lab study [150]. In order to do this, they had to write software which passed the eye tracking data to a JavaScript plugin as mouse data, and convert the text users would read to a webpage in which each word is encapsulated by span tags (and thus becomes a

unique item in the DOM). The JavaScript plugin then records each 'mouse' hover while a participant is reading. Both of these studies demonstrate that modern web tracking software cannot be used for all reading stimuli, and is not easy for researchers to use.

Tracking tools that integrate with offline programs have been developed for collecting keystroke [5, 91, 99, 159], mouse movements [5, 91, 99, 131, 155], scrolling and menu selections [5], and window manipulation [99]. In some cases these tools also collect data on the state of documents directly from the applications they are monitoring [91, 131].

The drawback of desktop action trackers is that only a few actions such as mouse movements and keystrokes can be tracked without developing the tracking tool specifically for the application. Even if these are the only actions a researcher cares about, unmodified applications give very little information about the files that are currently displayed, so it is not possible to link mouse movements to the text a reader is pointing to. Other researchers have tried to add this information. In 1998, Horvitz et al. worked with the Excel development team to create a version of Excel that could track mouse clicks, keystrokes and the data in a file [91]. In 2006, Oliver et al. developed a method for logging window switching behavior at the operating system level on a Windows computer [158]. They show that this information can be used to cluster applications and monitor work. In 2008, Alexander et al. built a tool that could track higher level actions such as scroll bar manipulation and menu selections on Windows computers without modifying the tracked applications [5]. However, without modifying the applications, they could not track the state of files in the application. For example, researchers wishing to track programming students as they edit code use modified IDEs [131, 159]. TESS allows researchers to track text locations without modifying the software used by participants.

None of the existing software that tracks interactions as they happen is suitable for studying digital reading because it is not easy to setup with all stimuli. Building a tool that can track interactions through a webpage is different from building a tool that tracks interactions in a Windows desktop applications which is different from building a tool for

tracking interactions on an iPad application. However, students are using all these applications regularly to read textbooks, and researchers who hope to compare their experiences need tracking software that can easily be applied to all three scenarios. The only way to do this is to use image processing, because video is not complicated by how a reading application works, it only shows what a human user sees. TESS tracks text colors and positions from a video feed that shows what the participant was looking at.

2.2.3 Optical Logging Software

The need for automated optical logging to research how user interact with applications has been discussed by several researchers [19, 60, 72]. However, none of the existing optical logging tools was built with a focus on reading research, and as a result, none are a good fit for this domain. Chang et al. proposed a system that analyzes GUI design from screen captured videos [48]. Dixon et al. have used similar methods to build tools that reverse engineer and edit GUIs to provide language support [60, 61, 62]. In 2016, Frisson et al. proposed a system that tracks buttons and windows defined by a researcher for studying general application usage [72]. Bao et al. have worked on using screen capture videos to track programmers as they use IDEs [19, 20, 21]. Ponzanelli et al. applied video tracking of IDEs to programming tutorial videos in order to make the tutorials interactive [169, 170]. All of these applications are important, but they do little for reading researchers, since most electronic reading environments, unlike the applications mentioned above, do not use button clicking or typing as the main mode of interaction with the user. In 2012, Banovic et al. attempted to build a general framework for extracting user interactions from screen captured videos [18]. However, the interactions they concentrate on are focused on buttons and mouse shapes that are often absent from digital reading environments such as dragging widgets. Reading experiments are concerned with where in a document participants are spending the most time [41, 140], what participants are looking at [38, 51, 112, 117, 118, 141, 148, 228], what participants are taking note of [30, 36, 154, 168] and how participants

are navigating [69, 73, 86, 140, 185, 193]. To track these activities, a tool must record when text moves and changes color.

Outside of screen recordings, a lot of work has been done on tracking movement in videos generally [223]. Most of this work is aimed at tracking movement in videos of the real world [76, 87, 103, 203]. Unfortunately, assumptions that hold for videos of the real world break down in screen recordings of text. If each object in a video has a set of uniquely identifying points that distinguish it from other objects in the frame, optical flow and object modeling can be used to track the same object across multiple frames [103, 223]. If a small number of objects move across a visually heterogeneous background, the moving objects can be detected by taking the difference between the pixels in a frame and the average value of a pixel across multiple frames [223]. In some screen recordings, such as when the user is playing a video game [134], objects on the screen conform to these assumptions. However, in screen recordings of text, each copy of a letter can be described using the same points, the background is usually monochrome, and the background usually moves with the text when a user scrolls. For these reasons, general movement tracking algorithms such as optical flow do not work for tracking user actions in reading experiments.

To successfully track words across frames, TESS uses OCR to find word objects in each frame. OCR is a method for finding text and its position in an image that has been well studied for many years, though building OCR engines that perform as well as humans is still an active area of research [100]. Previous work on improving OCR results has concentrated on adjusting images to make them easier for the computer to read [53] and using statistics and dictionaries to find likely mistakes in unknown text [64, 188]. The idea of using OCR to track text positions during a reading experiment was suggested by Buscher et al. in 2008 [40], but TESS is the first tool to use OCR for this purpose. TESS can be much more accurate than an unmodified OCR engine because it knows the text of the files participants are reading and only needs to find the position of each word on the screen.

There has been some work on using OCR to enable viewers to search videos of slide

deck presentations by the text on a slide [2, 190, 205, 219, 220, 221, 229]. Like TESS, these tools correct mistakes from the OCR results using the text of the slide deck. However, since the text is only used for navigating to video frames, the exact position of text within a frame is not calculated. As a result the correction algorithms employed do not drill down to the word level the way TESS does. TESS is the first tool to use OCR to track the exact location but not the text values of words in a video.

2.2.4 Similar Tools and Methods Summary

Human coding does not scale well and is not suitable for recording 'micro' actions. In cases where researchers are interested in subjective 'macro' actions, human coding of 'macro' actions is made faster and more accurate by visualizations of micro actions [43, 130, 163]. TESS provides these visualizations for digital reading studies. An evaluation of the usefulness of TESS's visualizations for macro reading actions is described in Section 4.7.

Tools that track interactions as they happen by interfacing with stimuli software must be customized to the stimuli software, which takes too long in a world of constantly changing reading environments. For some experiments, like testing search engine strategies, this works because the research team either writes the stimuli software themselves or uses software that doesn't change very often. Educational reading researchers run experiments on the tools that are currently being used in schools, which schools often change in response to private sector lobbying, policy changes, or student requests [37]. To quickly study digital reading environments that are already being used, researchers need tools that can analyze student interactions with new software and don't require researchers to re-engineer tools.

So far, no other image processing tools track 'micro' actions for digital reading environments. Actions such as scrolling, highlighting, and dwelling on text have been shown to indicate what readers are taking away from an article [86, 122, 213]. Tracking these actions is necessary for designing better books, and educating the next generation.

2.3 Related Research Summary

The way students read is changing quickly. Print books do not change after publication, but every interaction with an electronic book can be modified by software updates that are automatically installed while readers sleep [71]. This means that research findings can be applied to improve books immediately instead of waiting for the next publication cycle, but it also means that researchers have a smaller window for setting up and running a study if they want to observe the reading environments students are currently using.

TESS allows researchers to quickly setup studies. It works for all digital reading environments, and can easily be applied to a large data set of screen recordings. The following chapters describe how TESS works, describe evaluation tests and their results and present case studies of TESS used in reading experiments.

Chapter 3

System Design

As described in Chapter 2.1, TESS is designed to track text movements and interactions in all digital reading stimuli at scale. It does this by reasoning about the visible text in screen recordings of study participants reading. Note that TESS requires researchers to input the corpus users read from and the possible colors of highlights in addition to the screen recording videos. These inputs allow TESS to generate precise and accurate outputs. In future work it may be interesting to explore the value of less accurate outputs from a tool that only uses the screen recording videos. For a full discussion of extensions to TESS see Chapter 7.

TESS divides each participant's screen recording into time windows when the screen was static, then OCR is run on a frame from each time window. In most reading experiments, the screen routinely remains static for several seconds at a time while the participant reads the display. TESS goes through the OCR results and corrects the text labels using text files that describe the study stimuli and categorizes each detected word according to its highlight color. Other data sources such as gaze data can then be paired with the corrected results to find the words participants fixate on. Actions such as scrolling and highlighting are detected by comparing the corrected results for consecutive time windows. Any remaining errors can be corrected by human editors using an interactive editing module. Section 3.6 discusses how TESS's output can be iterative improved by using the interactive editing module and rerunning some of the steps. A flow chart visualizing this process is shown in Figure 3.1. Each step of the pipeline is described in greater detail in the following sections. The code for TESS can be found at github.com/kbrady/tess.

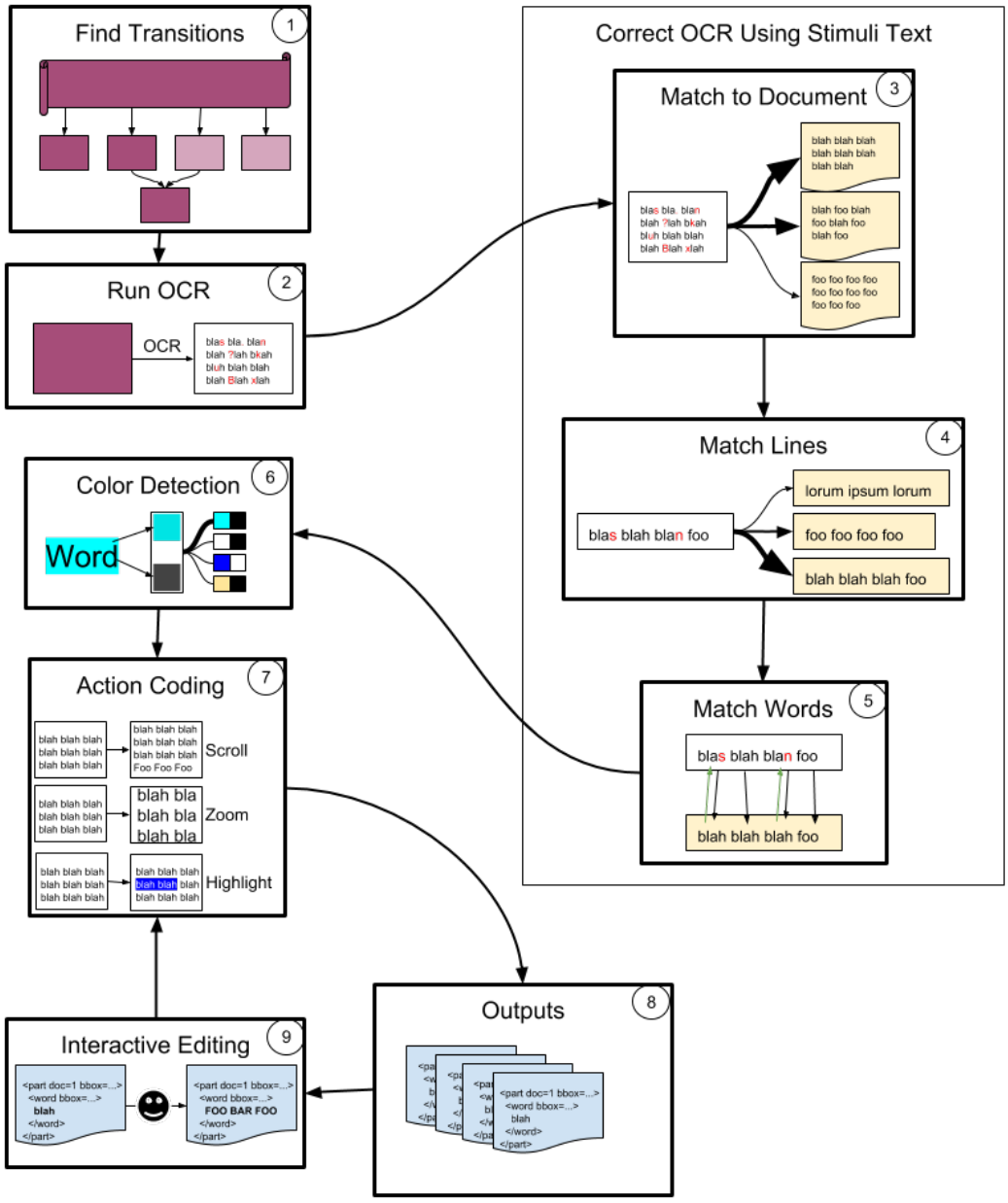


Figure 3.1: An overview of TESS's process of extracting information from a screen recording.

3.1 Finding Transitions

The first step in the pipeline is to find periods of time in the screen recording when the screen changed. Processing each frame takes a long time, so TESS's goal in this step is to find the minimum set of frames which capture all periods in which the screen changed.

Generally participants have short bursts of highlighting or scrolling during a reading experiment followed by long periods in which they read the display without changing anything on the screen.

It is more important that TESS identify every time that the screen changed than that it not identify a change when one did not happen. The negative effect of identifying an extra change is that TESS runs slower, while the negative effect of not identifying a change is that data is lost. A good solution will still ignore most frames since most readers scroll and highlight only a few times per minute. The number of frames per second in a screen grab video will vary with the software used to capture it, but it is typical to have at least 30 frames per second. Examining every frame in a reading session is time intensive and unnecessary since even if an experiment is tracking gaze, humans only have about 5 fixations per second [176]. Other actions such as typing and clicking take longer.

To find static time windows, TESS uses a modified binary search. It pulls out each pair of frames which differ by time T, where T is small enough that it is unlikely the user took an action and undid it in T. If there is a difference between the two frames, TESS does a binary search of the time stamps between the frames to find the time positions when the screen changed. There may be multiple positions when the screen changed (particularly if the change corresponds to a scroll), so each time TESS finds a pair of non-matching frames it conducts a binary search between them. The search stops when the difference between frame times falls below a threshold t.

The finding transitions step is illustrated in Figure 3.2. Since the frames found at 9:30 and 9:40 are the same, TESS doesn't look at any frames between them. The frames found at 9:20 and 9:30 are not the same so TESS continues searching for the time of the transition. The frame at time 9:25 matches that at time 9:20, so the search continues between 9:25 and 9:30 but stops between 9:20 and 9:25. If 9:25 had matched neither the frame at 9:20 nor the frame at 9:30, TESS would conduct two binary searches, one between 9:20 and 9:25 and one between 9:25 and 9:30. Once all the transition times are found, only one of the frames

in each time window will be used for future analysis. In this example, 9:20 and 9:30 might be kept to represent two views of the screen, but the image of 9:40 will be discarded.

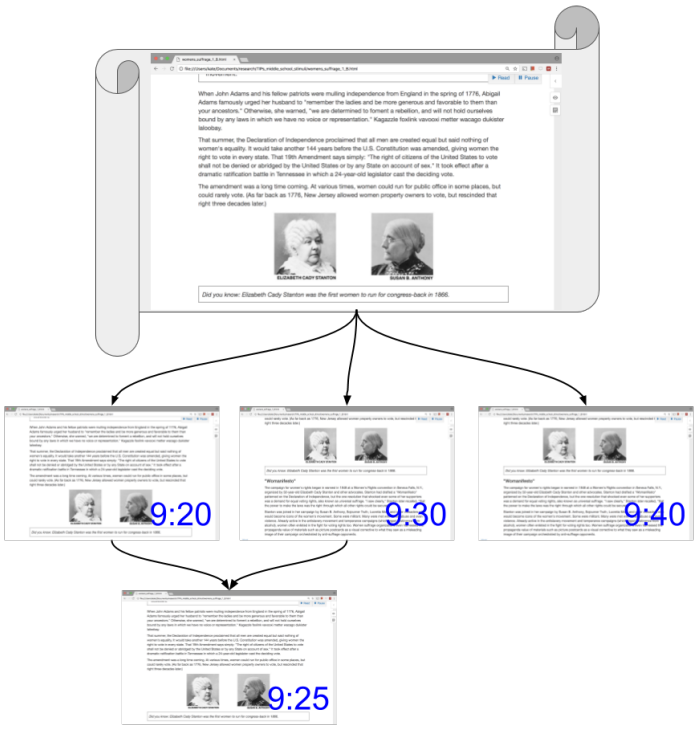


Figure 3.2: The process by which times when the screen changed are found.

This works as long as the user doesn't change the screen and change it back again in time T, and doesn't make any meaningful actions in less than t. For optimal performance, researchers should use the largest values of T and t that they can be reasonably certain these assumptions hold for. If pop-ups open and close quickly in the stimuli, or users briefly select text without saving highlights, then T should be only a second or two. When pop-ups are not part of the stimuli, T can be longer. Studies should set t in accordance with the user actions that are examined. For example, if a study is looking at eye fixations, then .2 seconds is a good value for t since that is the length of a fast gaze fixation [176]. Since TESS is run after screen recordings are collected, researchers can review some sample videos to check the length of actions before running TESS.

Screen recordings do not have as much noise as videos taken with a camera, since environmental noise does not affect graphics cards the way it does camera sensors. However,

screen recordings still contain pixel fluctuations that are invisible to the human eye. An example of one of these fluctuations is shown in Figure 3.3. Noise in images recorded by cameras usually occurs cyclically throughout the image due to the nature of how light moves. Since the noise in screen recordings is not caused by light movements, the noise reduction algorithms developed for cameras are not applicable to reducing the noise here. When finding frame differences, TESS instead filters out false positives by requiring changes to cover an area about the size of a word. None of the actions performed during reading cause single pixels sprinkled around the screen to change at the same time, so this filter ensures that the changes seen by TESS are consistent with the actions that researchers are looking for.

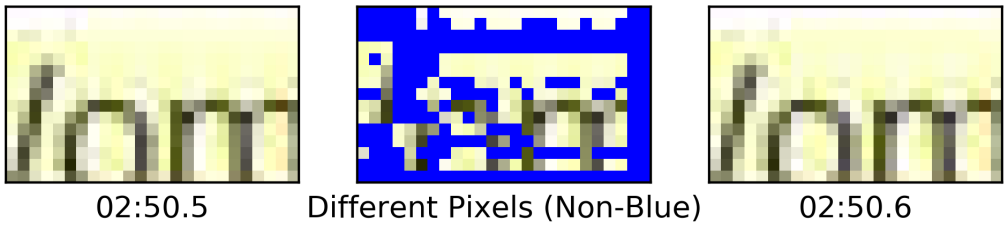


Figure 3.3: An example of a pair of images that look the same to a human but have different pixel values. The center image shows the pixels in the first image which differ by at least 20% of the color spectrum (50 out of 256 pixel values) in one of the three color channels from the pixels in the final image. The unshaded pixels all differ by at least this much.

Two frames are considered different if at least 90% of the pixels differ in a square area that is twice as long as the text height. This metric filters out small fluctuations between frames of the screen recording like those shown in Figure 3.3. The height of the text can be calculated by running the OCR engine on a sample frame and taking the median height of bounding boxes in the sample. An evaluation of TESS's finding transitions step is described in Section 4.1.

3.2 Finding Text

TESS tracks text positions over time in a screen recording by finding the positions of words in each frame of the recording. Once a frame has been selected by the transition finding step as representative of a time interval, TESS runs the finding text step to extract the positions of each word in the frame. This step first enlarges the frame to make it easier for OCR engines to read. A test of the enlarging images step is described in Section 4.2. Then TESS uses an OCR engine to extract text information from the frame and store it in an hOCR file, and finally makes corrections to the hOCR file by comparing the results to the text of stimulus used in the experiment. Sections 4.3 and 4.4 describe tests that evaluate the correction algorithm. The result of the finding text step is an hOCR file for each time window that was selected by the finding transitions step. This file contains information on the position and text value of every word displayed on the screen in that time window.

In 2007, hOCR was designed to be an open standard for OCR engines to describe their output [34]. It has been adopted [216] by Tesseract [194], OCRopus [35], and Cuneiform/OpenOCR. These engines are all freely available and open source [189]. Other standards have been developed for representing OCR results [215], but have not been as widely adopted. One reason hOCR has been successful is that it is an extension of HTML (Hypertext Markup Language). As a result, hOCR documents can be viewed in any web-browser and parsed by any HTML reader or library.

One of the goals for TESS discussed in chapter 2.1 is that TESS be easy for researchers to setup. Using the hOCR standard allows researchers to swap in any OCR engine that outputs hOCR files if they have access to state of the art technology. For researchers who do not have access to commercial OCR engines, TESS is designed to work with the most widely available open source OCR engines. Tesseract [194] was used as the OCR engine when building and testing TESS.

TESS prepares frames before running OCR by enlarging the image. This step improves the initial OCR results because Tesseract works best when characters are at least 20 pixels

tall [217]. Once the frame image has been prepared, the OCR engine is run on the frame and the output is saved to a hOCR file that is named for the time window represented by the frame. For a discussion of the trade offs between quality OCR results and the time needed to adjust the image, see Section 4.4.

TESS is designed for experiments in which the text being read is part of a corpus of preset stimuli. This constraint allows TESS to correct errors by matching words recognized by the OCR engine to the most similar text sections in the stimuli corpus and replacing the incorrect text. This matching is performed hierarchically by first matching the hOCR file to a document in the stimuli corpus, then matching the lines in the hOCR file to lines in the stimuli file, and finally matching words within matched lines to each other. This process is shown in the 'Correct OCR Using Stimuli Text' box in Figure 3.1. The following subsections explain each step of this hierarchy in more detail.

3.2.1 Matching Frames to Documents

TESS matches hOCR files to documents by calculating a similarity score, $S(f,d)$, between each document d and the hOCR file f. The hOCR file is matched to the document with the highest similarity score. The similarity score is based on the number of words which appear both in the hOCR file and the corpus document and their length. The equation for $S(f,d)$ is shown below:

$$S(f,d) = \sum_{token \in (f \cap d)} length(token)$$

This matching algorithm assumes that documents in the stimuli corpus contain different words and that a set of words unique to the correct document will be correctly recognized by the OCR engine. The first part of this assumption will depend on the reading material, but will hold for the stimuli used in a lot of reading studies. In a sample of 80 randomly chosen English Wikipedia articles, 1-14% of the tokens in each document are unique to

that document, and 4-26% were in less than four other documents. This means that using a bag of words model, a screen grab with only 25 words from a document should have at least one word matching it to only 5 out of 80 documents. By considering the evidence from more than one word, TESS's matching algorithm does better than this.

The second assumption, that a set of words unique to the correct document will be correctly recognized by the OCR engine, will hold if some words in each frame are recognized correctly and words that are given incorrect labels by the OCR engine only marginally increase the similarity score between an hOCR file and the wrong document. The assumption that at least some words in a frame with more than 10 words will be identified correctly is supported by tests of widely available OCR engines like Tesseract. Tests of Tesseract have found that it correctly identifies 36-94% of words depending on the script and language [195]. It has been tested on Latin, Cyrillic, Arabic, Indic, Chinese, Japanese, Thai and Korean scripts. Tesseract's worst performance is on Indic scripts and it works best on Latin scripts. The accuracy for any language can be improved by adding a dictionary, but does not rise above 94.3% without more information about the text.

There is no way for the document matching algorithm to know which words have been correctly labeled by the OCR engine and which have not. The similarity score attempts to prevent incorrectly identified words from raising the similarity between an hOCR file and an incorrect document by weighting longer words as more important than shorter words. The probability that a small number of letter changes will result in another valid word decreases as words become longer, since the number of possible letter strings grows exponentially with word length, while language vocabularies do not.

In many languages, short words are also more likely to be common throughout a corpus. An analysis of 11 languages showed that on average stop word are only 30%-65% as long as the average word in the language's dictionary [29, 162]. In a corpus of 80 Enlgish Wikipedia articles, the median word with three characters or less was off by one letter from 15 other words. In contrast 44% of words with four characters or more were off by one

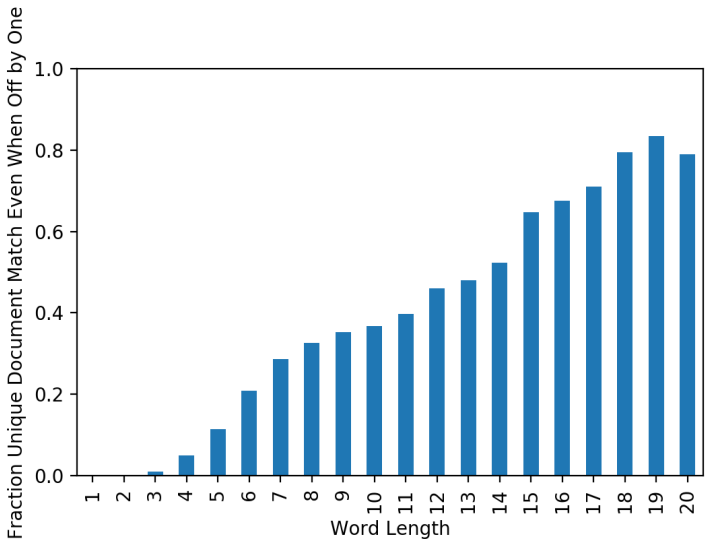


Figure 3.4: The fraction of tokens that uniquely match one document, even if a character is changed based on length in a corpus of 80 English Wikipedia documents

letter from no other words in the corpus. Longer words are more likely to be specific to the correct document. In the Wikipedia corpus, 44% of words with three characters or less were unique to one document. Fifty two percent of words with at least 4 characters match only one document. Combining these statistics reveals an almost linear relationship between the length of a word and the probability that even if a match was off by one letter, only one document would be matched. This relationship is illustrated in Figure 3.4. To build this into the model, TESS weights the evidence of a match by the matched word's length. An evaluation of TESS's document matching step is described in Section 4.3.

3.2.2 Line Matching

Lines are matched by first using a similarity score like that used for document matching. If a line does not have a sufficiently strong match with this similarity score, TESS uses the Levenshtein edit distance [188] to find its match. The similarity metric used for matching lines looks at the words that occur only once in the displayed document. If a line detected by the OCR engine and a line in the stimuli text file share two of these words, the lines are considered a match. TESS uses a threshold above 1 word to avoid matching lines on

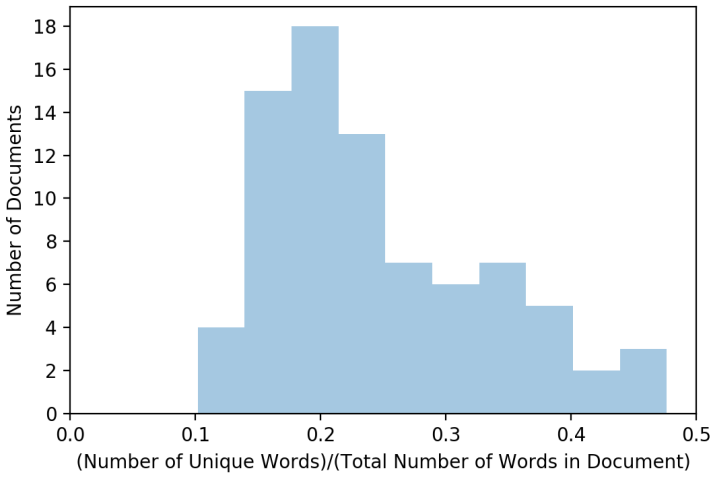


Figure 3.5: This figure shows the number of documents in a Wikipedia corpus that have various probabilities of sampling a word unique to the document. The x-axis shows the probability a sampled word occurs exactly once in a document. The y-axis shows the number of documents in the corpus with that probability.

a word that was recognized incorrectly by the OCR. In a sample of 80 English Wikipedia documents, 10-48% of words in each document occured exactly once. The distribution of unique words in this sample is shown in Figure 3.5. Based on this analysis, a line of text with 20 words should on average contain 2-10 unique words.

Once lines that can be matched using the similarity metric have been paired, this pairing is used to restrict the search space for the remaining lines. Lines in the hOCR file that have not been matched are compared to unmatched lines in the text file that occur between the same pair of matched lines. In experiments, the number of unmatched lines during this step has been small. Most unmatched lines result from either the hOCR or text file having more lines. This can occur when the OCR detects extra lines or when a line is covered by a pop-up. In both of these cases, the correct result for the unmatched line is to delete it.

The proper matching between unmatched lines is determined by calculating the Levenshtein edit distance and choosing the configuration with the smallest number of edits. Edits that involve deleting characters from the ends of each line are weighted as less important than edits that change the middle of a line, since characters will disappear from the screen

if the user zooms in or opens a sidebar. If the weighted Levenshtein edit distance between a OCR detected line and its assigned correction string in the final configuration is less than 60% the length of the detected line, the string is used to correct the line. Otherwise, the line is marked as not part of the document, and is left unmatched. If a line is unmatched at the end of this step, it is deleted. An evaluation of the improvement in text labels after line and word matching is described in Section 4.4.

3.2.3 Word Matching

TESS matches words by starting with an initial solution based on character widths and then improving upon it. Character widths are calculated from the words that were used to calculate the pairs in the first step of line matching (henceforth referred to as 'line words'). These words are presumed to be correct, since they exactly matched a unique word (presumably from the correct line) in the text document. The width of each character in pixels is calculated by summing the widths of all 'line words' that contain that character and dividing by the number of characters in that subset. Based on this estimate, words are paired according to their expected distance from a starting word. A few values are used for the starting word, and the final pairing is the one that has the lowest edit distance after hill climbing. Each word in the corpus document is paired to the word in the hOCR file that has the most overlap with its expected position (based on the character width estimate). If no words overlap, then the initial matching marks the word as 'not visible', and therefore without a match.

Once the initial matching is set, TESS calculates the sum of the Levenshtein edit distance between each word and its pair. TESS then attempts to decrease this value by making incremental edits to the pairing. These incremental edits include shifting the matching over by one word and shifting a sub-sequence over by one word. TESS checks each possible incremental change and adjusts the matching to the change that lowers the edit distance the most. When none of the available changes have a smaller edit distance than the current

matching, the matching is finalized. At this point, the text of each word in the hOCR file is replaced with the text from the matched word in the text stimuli. The original text from the OCR is saved as an attribute of each word in the edited hOCR file for debugging purposes. An evaluation of the improvement in text labels after line and word matching is discussed in Section 4.4.

3.3 Word Color Detection

Once the location of each word in a frame has been discovered, the colors of the pixels around that word can be used to determine whether it was highlighted and what color highlight was used. The list of possible highlight colors (including the default color) is defined by the researcher as a list of color pairs that represent the text and background colors. This list helps TESS group highlights that may appear as different colors to the computer due to noisy pixels.

TESS detects the highlight color of each word in a frame by partitioning the pixels in the region the word is detected into two groups. It then takes the median of each group and compares these values the list of possible color pairs set by the researcher. The area of pixels examined is a few pixels larger than the bounding box found by the OCR engine. This ensures that enough background pixels are included in the analysis, even if the word being examined is only one letter long and does not include any background pixels in its bounding box. How much the larger the examined area should be than the bounding box can be adjusted based on the size of the text and how far the highlight color extends around each word. In evaluation experiments, a border of three pixels around the bounding box worked best for most highlighting tools.

TESS has two methods of dividing pixels into two groups. The first method is faster, but does not work for all color pairs. The second method works for all distinguishable color pairs but takes longer to compute. TESS can determine whether the faster method will work automatically by measuring the gap between average brightness values for color

pairs in the highlight colors list.

If all the possible highlight and text pairs include one light color and one dark color, then partitioning pixels can be done quickly using a threshold. For each pixel, the average brightness across the color bands is computed. If this value is greater than the threshold then the pixel is added to the 'light' group, otherwise it is added to the 'dark' group. This method of dividing pixels is fast since pixels can be processed in parallel.

In some cases, both colors in one pair will be above a brightness threshold and both colors in another pair will be bellow that threshold. In these cases, k-means clustering can be used to separate the pixels into two clusters while taking all three color channels into account. K-means clustering is a sequential algorithm, so this method takes longer than the threshold method. Evaluation tests of TESS's word color detection method using both thresholding and clustering are described in Section 4.5.

3.4 Action Coding

Once the text positions and colors have been found in each frame, it is still non-trivial to calculate actions across frames. To help researchers looking at common actions, TESS has modules which aggregate scrolling, zooming and highlighting data. These modules find the position of the top and bottom of the screen relative to an 'image' of the entire document to represent scrolling and zooming. The highlighting action detection module finds instances when the color of text changed and creates a report of the times when a change was detected and the text that changed. Evaluations of TESS's ability to correctly aggregate scrolling and highlighting actions are described in Section 4.6.

3.4.1 Scrolling and Zooming

Scrolling and zooming can be modeled as moving and re-sizing a viewing window over a large image of the entire document. The positions of the lines in each frame can be used to project the frame onto this theoretical 'image'. If the frame contains lines that are closer

together in the ‘image’, that indicates the reader zoomed in. If the frame contains lines which are lower or higher in the ‘image’, the reader scrolled down or up. The positions of lines in this theoretical ‘image’ can be calculated by stitching together the frames analyzed by TESS. Once the aggregated ‘image’ is stitched together, the position of individual frames within the image is calculated.

In some cases, such as when a reader jumps around an article using links and does not view all lines in the article, it may be helpful to make a real image by stitching together screen grabs of the article. TESS can then be run on the real image and the detected line positions can be substituted for the values in the stitched ‘image’. Note that since scrolling and zooming are always measured in terms of previous positions, it does not matter if the image used to model the document has larger or smaller text than the screen recording being analyzed. Real stitched images can also be helpful for giving context when visualizing scrolling and zooming, as demonstrated in Figure 3.7. The drawback of making a real stitched image is the time and effort required. In cases where the reader viewed all lines in the article, it is easier to let TESS build a stitched ‘image’ automatically. The rest of this section describes how TESS does this and how the line positions are used.

The position of each line in the stitched ‘image’ is calculated from the positions observed in the frames. First, all instances of the first line in the article are found in the corpus. The top and bottom values of the first line in the stitched together ‘image’ are equal to the average observed top and bottom values of this line in the corpus. Next, all instances where the first and second lines were both observed are collected. Each observation is scaled so that the height of the first line is the same as the height of the first line in the stitched together ‘image’. Then the distance from the top of the first line to the top and bottom of the second line is calculated in each observation. The top and bottom of the second line in the stitched ‘image’ are assigned to the top of the first line plus the average distances in the observations. This process is continued with the second and third lines, then the third and fourth lines, etc. until all lines positions have been added to the stitched

'image'.

If two participants read the same article, but one zooms in, the one who zoomed in will have larger pixel values in their stitched 'image'. This is caused by larger values for the top and bottom of the first line, which then propigate to all other line measurements. For this reason, when comparing scrolling between two users reading the same document, the scrolling measurements should be normalized by the distance from the top to the bottom of the stitched 'image'.

Once the list of line positions has been calculated, a projection is calculated between each frame and the 'image' by finding the positions of two lines in the frame and two lines in the 'image'. The top and bottom lines in the frame are the most likely to be incorrect since these are the lines that get partially cut off most often and thus have the worst OCR readings. The central lines are more likely to have the correct labels, but small errors in the positions of these points will propagate to large errors in the projection since the points are closer together. To balance these issues and get the most correct projection possible, TESS calculates a projection based on every pair of lines in the frame. It then takes the median of the shift and multiplication values that describe each projection and builds a projection with these median values.

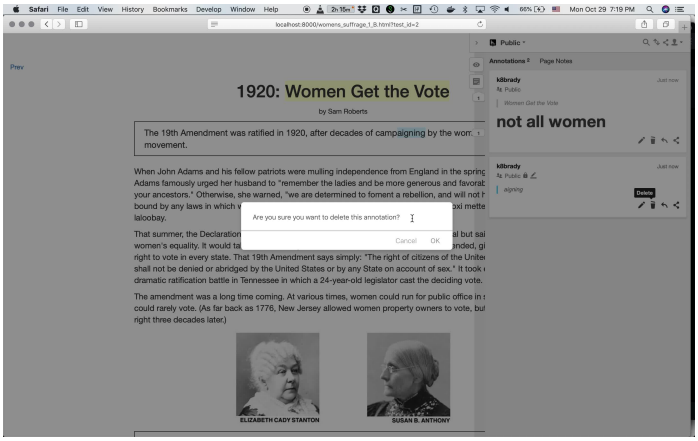


Figure 3.6: A frame in which the OCR engine failed to recognize text from the article because the background was too dark.

If less than four lines are detected in a frame, then TESS assumes that the user navigated

to a document outside the corpus or zoomed in too much to get a reliable reading. The frame position for these frames is either replaced with the position of a previous frame or set to null depending on the preference of the researcher. Using the previous frame may be the right choice if frames are likely to become unreadable do to pop-ups (such as in Figure 3.6) while setting the scroll value to null may make more sense if users are frequently navigating to files outside of the corpus.

After TESS settles on a projection for a frame, the top and bottom of the frame are calculated by applying the projection to zero and the height in pixels of the frame. These two values are saved for each time window. If a user zooms in, the top and bottom values move closer to each other. If a user scrolls down, the top and bottom values increase at the same rate. Scrolling and zooming are both based on the vertical position of each line. As a consequence, it is not necessary to calculate word positions for studies of scrolling and zooming. Researchers who are not interested in word based metrics can save time by only running the document matching and line matching steps of the correction algorithm and then running the scrolling detection module.

3.4.2 Highlight Coding

The highlight detection step described in section 3.3 finds the color of each word in a frame. In many cases, researchers are interested in when words change color, which requires aggregating color information across time windows. TESS does this by assigning each detected word a global id which is the same across all time windows. If a word block detected by the OCR engine contains multiple words, then that block will be assigned multiple global ids. Global ids are assigned during the action coding step rather than during the text correction step so that they can be updated if word labels are manually edited.

Highlighting actions are aggregated by going through the frames in order and noting each time a word changes color. A hash map is initialized with white as the color of each word. If a word appears in a frame and its color is not the same as in the hash map, the

time of the frame, the word's global id, the former color of the word and the new color of the word are added to the highlighting record. Then the color of the word in the hash map is updated to the new color.

If a word block detected by the OCR engine contains multiple words, the detected color of the word block must differ from the stored colors of each constituent word in the hash map for the observation to be recorded as a color change in the highlight record. If any of the words in the block already had the detected color, TESS assumes that the color of the whole block was determined by that word. As mentioned in section 3.3, this is a shortcoming of labeling each word block only one color.

3.5 Outputs

TESS's output can be paired with other tracking data, aggregated into 'mico' actions or visualized to find 'macro' actions. As a result, TESS has three types of outputs:

1. A corpus of hOCR files
2. Aggregated Action Files
3. Visualizations

The corpus of hOCR files is good for merging with other time series data sources, such as mouse movements, gaze data, and emotional response data. This output gives the positions, sizes and highlight colors of words at all times during a screen recording. Linked to other time series data, this information can tell researchers what words readers fixated on as they scrolled and what words they hovered over with the mouse. This form of output is also large and difficult to reason about. For this reason, aggregating or visualizing the output is often necessary for relating the output to a hypothesis.

TESS has modules to aggregate its output for scrolling, zooming and highlighting, which are common actions that researchers have already expressed an interest in. These

aggregated outputs are stored in single files that are easier to work with than the hOCR corpus and use more common data standards. This makes it easy to import the aggregated output files into statistical modeling software and quickly generate histograms. Researchers who plan to use the corpus of hOCR files to closely analyze their data can first look at the aggregated data files to get an overview of the data and check for obvious errors.

Some 'macro' actions are difficult to describe in terms of 'micro' actions, but are understood by human coders. In these cases, TESS's output can be visualized to assist human coders. Human coders can use these visualizations to label participant actions faster. In studies where there are too many participant videos for human coders to label them all, human coders can use visualizations to quickly label a subset of participants and then train a computational model to label the rest. Visualizations of the data can also make some actions easier to see. For example observing how many times text was selected and unselected is easier when time is shown as an axis.

3.5.1 Corpus of hOCR files

TESS's default output is a corpus of hOCR files that are named according to the time of the frame they represent. Each file contains information on the position, text label and highlight color of each word in the frame. The files also store information on how these values were calculated and whether it is likely that a mistake has been made. For example, if the text label assigned by the OCR differs radically from the corrected text label it is more likely that the text label is wrong. Additionally, if the medians of the color partition and the assigned highlight color pair have a large gap, the detected highlight is more likely to be wrong. Each hOCR file can be parsed by any XML (Extensible Markup Language) or HTML interpreter. The data in the files can be paired with other data sources that incorporate time and screen coordinates as discussed in Section 2.1.5.

Researchers interested in tracking the amount of screen time each word, line or paragraph received during a participant's session can define use the corpus of hOCR files to

compute which time windows the text of interest was displayed in. These calculations will be more accurate when they are made at the line level than at the word level since the hierarchical nature of the text correction algorithm prevents the word corrections from being more correct than the line corrections. A discussion of the value of dwell time data in reading research is presented in Section 2.1.3.

3.5.2 Aggregated Action Files

In cases where researchers are not linking other data sources, the hOCR corpus may be overwhelming to work with. In these cases researchers may prefer to work with the outputs from the action coding step. The scrolling/zooming action coding step outputs two CSV (Comma Separated Values) files. The first describes the stitched 'image' that all frames are projected onto and the second describes the position of the top and bottom of the screen on this projection for each time window. The columns of the CSV file describing the stitched image are the text of each line, the top position of the line, and the bottom position of the line. The columns of the CSV file describing the screen's position over time are the start of each time window, the top position of the screen on the stitched 'image' during that window, and the bottom position of the screen on the stitched 'image' during that window.

Researchers can open these CSV files in any program that reads spreadsheets. They can compute how much the user zooms in by looking at the distance between the top and bottom of the screen over time. To compare scrolling and zooming behavior between users, it is advisable to scale the results by the distance from the top line to the bottom line of the stitched 'image' since this value will be bigger for users who zoomed in.

The highlight action coding step outputs a JSON (JavaScript Object Notation) file that lists all the times that a color change was detected and what the color changes at that time were. Each word block that changed color is represented by its word labels, global ids, new color, and the previous colors of the words in the block. Generally word blocks only contain one word, but in some cases the OCR engine fails to find word boundaries and

multiple words are put in the same block. Section 4.5 presents an analysis of how color plays a role in OCR detection of word separations.

The JSON file type was used rather than a spreadsheet or XML format because JSON is still human readable but has less redundancy for hierarchical data. The file groups color changes by time, and then stores the label, global id, old color and new color of each word that changed color at a given time. Researchers who wish to work with the data in a spreadsheet program can convert the file to a CSV by making each word color change a row.

3.5.3 Visualizations

Researchers interested in macro actions may find it difficult to describe those actions in terms of the 'micro' actions coded by TESS. To assist these researchers, TESS has modules to visualize scrolling, zooming and highlighting. Human coders can view these visualizations to manually code subjective 'macro' actions without watching the screen recording video. The visualizations also make it easier to spot likely errors in TESS's results. Using visualizations for error spotting is described further in section 3.6.

The visualization of scrolling and zooming shows the top and bottom positions of the screen over time. Time is plotted on the x-axis and a blue band showing the span from the top to the bottom on the stitched 'image' is plotted in the y-axis. The numerical values of this axis are the position of the screen on a theoretical image, which is meaningless for most applications, so numbers are not placed on the axis. If the researcher has a real image of the article stitched together, they can put this image behind the graph to give context to the y-axis (as shown in Figure 3.7).

Highlighting is visualized by plotting each word in the article in chronological order on the x-axis and plotting time on the y-axis. The color of a point on the plot indicates the color of the word indicated on the x-axis at the time indicated on the y-axis. If a highlight is kept by the participant, the word will stay the highlighted color and be visualized as a

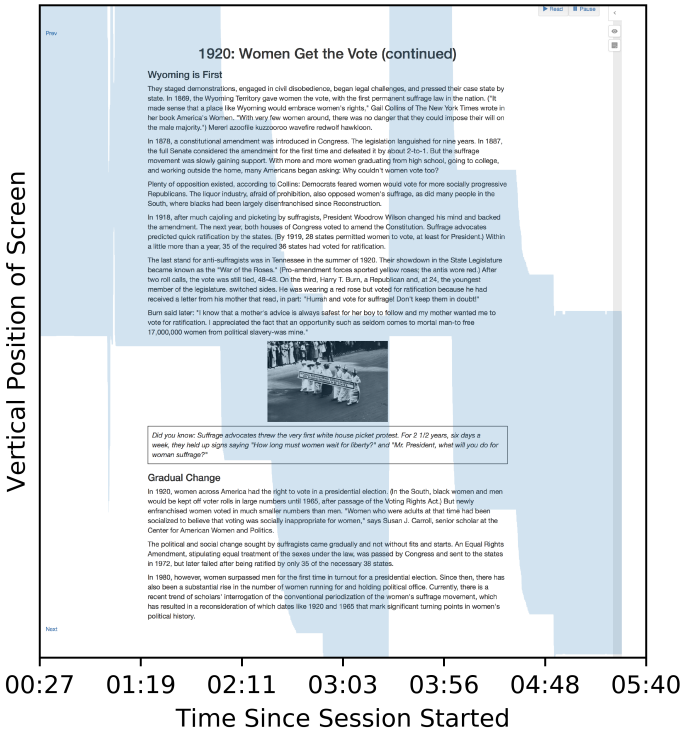


Figure 3.7: An example visualization of scrolling behavior. The image behind the graph is a 'stitched image' and provides context for the vertical position of the screen. Note that unless the user zooms in, they will see the full width of the 'stitched image' at all times.

vertical line from the point the highlight was made to the end of the session. The tick marks on the x-axis show the labels of a few words around that position to orient human coders. An example highlight visualization is shown in Figure 3.8.

One example of a subjective 'macro' action is re-reading for review. This action is difficult to describe in terms of 'micro' actions because it is dependent on a host of factors. Re-reading for review is defined as any instance where after the participant scrolls down slowly enough that it is plausible they were reading, they quickly scroll up and then slowly scroll down again. Here scrolling 'slowly' can include several instances of moving the screen quickly provided each line gets enough display time to be read. A computer coding this action would either need someone to precisely define thresholds for 'slow' and 'fast' scrolling or a large number of examples of reviewing and not reviewing scrolling behaviors to learn from. A human coder can look at the example in Figure 3.7 and determine that the

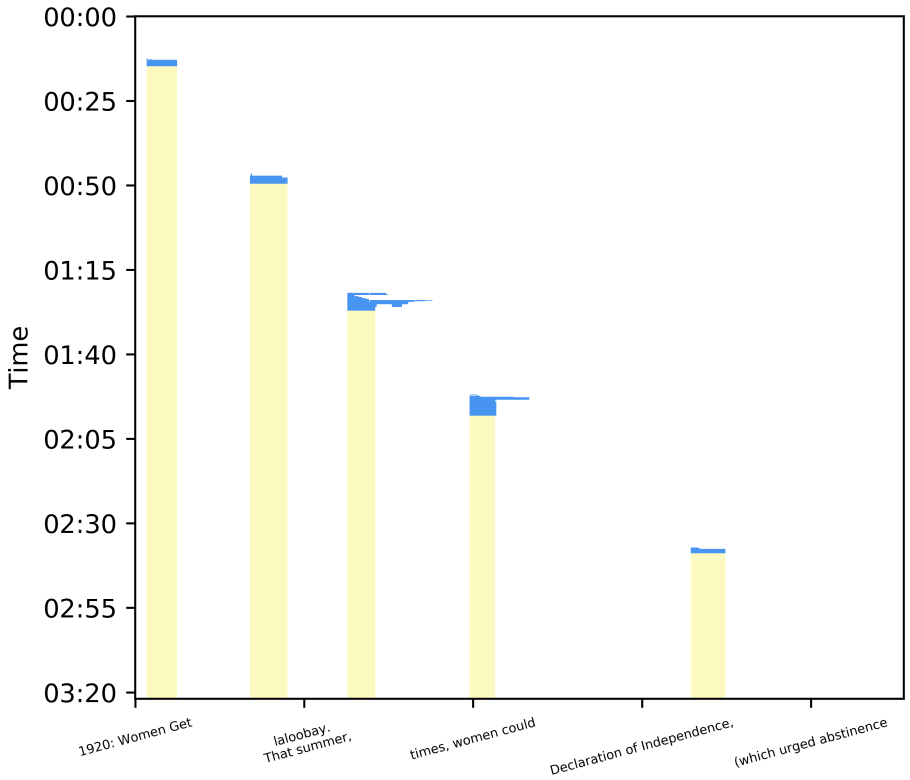


Figure 3.8: An example visualization of highlighting behavior

reader reviewed the text starting around 3:30 based just on the description of reviewing.

These visualizations are not meant to convey the exact word highlighted or number of pixels scrolled. The aggregated action files contain this information for researchers who are interested. The visualizations shown here can be used with the output files described above to find time windows in which TESS may have made mistakes or detected particularly interesting information. An evaluation of the usefulness of these visualizations for coding subjective 'macro' behaviors is described in Section 4.7.

3.6 Human-TESS Teaming

TESS, like most OCR based systems is not 100% reliable, but researchers can fix mistakes by interactively editing the output files and re-running the action coding step. This loop is illustrated in steps 7, 8 and 9 of Figure 3.1. Human coders can identify likely

mistakes by reviewing visualizations of TESS's output. For example, in Figure 3.9, TESS shows that the participant zoomed out quite a bit in the time window highlighted by the red box. This measurement is suspicious since digital reading environments usually don't let the user zoom out passed the initial zoom level. To fix this mistake, a researcher can use TESS's interactive editor to fix the frames in the relevant time window and adjust the line matching. A screen shot of this editor is shown in Figure 3.10.

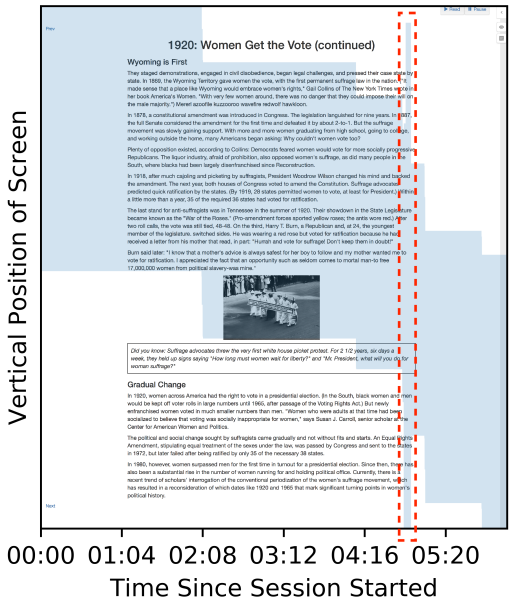


Figure 3.9: A visualization of TESS's scrolling results which shows the user zooming out 4 minutes and 47 seconds into the session. The screen usually starts maximally zoomed out, so this zoom is likely to be an incorrect reading.

Note that in the six minute reading session the incorrect reading represents only a one second window and can be spotted quickly by a researcher who knows what to look for. If mistakes are rare, easy to spot, and easy to fix, partnerships between TESS and human coders can create very precise reliable results. To facilitate human editors, TESS has an interactive editor that overlays information from each hOCR file on the frame it codes. Each detected word is indicated by a box around the word that is colored according to the detected highlight color. The label assigned to the word is displayed just above the box.

The color used to indicate a highlight is not the same as the highlight color since using the same color makes it difficult to see the box. A legend is shown in the top left corner of the page that indicates the mapping between box colors and highlight colors. Human editors can click on word boxes to edit their labels and highlight color assignments. A screen shot of this interactive editor is shown in Figure 3.10.

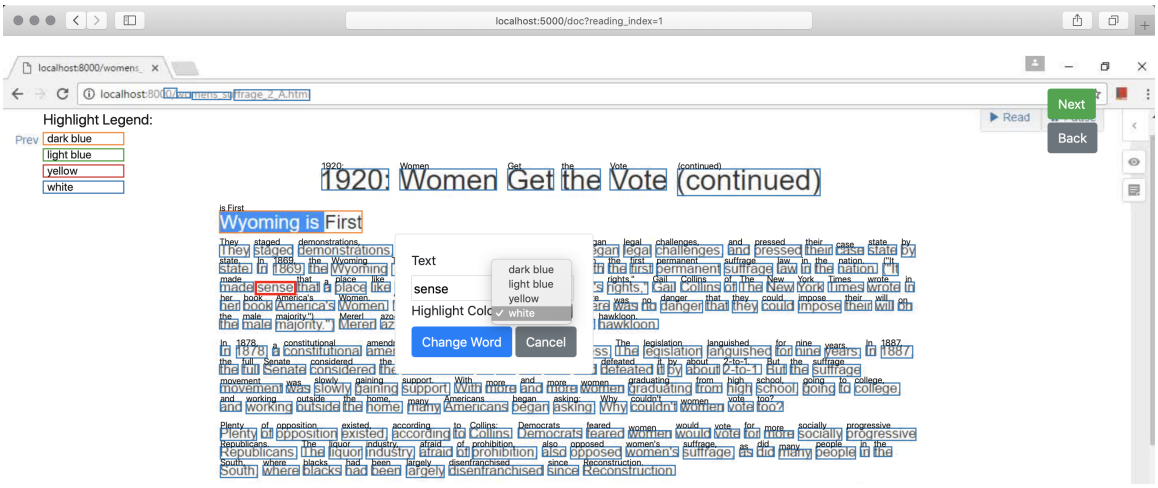


Figure 3.10: A screen shot of TESS's interactive editor. The word currently being edited is highlighted with a thick red box.

The time windows that are displayed in the interactive editor can be set by the researcher according to their timestamps or attributes of the hOCR output. In some studies, researchers are only interested in highlighting behavior or the words that a user looked at. In these cases a human editor can save time by only fixing the words that were highlighted or were displayed near fixation points in the gaze data. The interactive editor was used to mark whether or not TESS's output was correct in evaluation tests described in Sections 4.4 and 4.5.

3.7 System Design Summary

TESS has a modular design that enables researchers to adapt the tool to their needs. Researchers who only want to track document switching can stop TESS after running the document matching module. Researchers who wish to detect highlights in a stack of paper

documents can scan the documents to turn them into images and run the modules for finding text and colors without the finding transitions step. This modular design also makes TESS easier to evaluate and debug, as shown in Chapter 4. In future work, TESS's mortality will hopefully make it easier to improve.

This tool is designed to be easy to setup for any reading stimuli, track text movements, and scale for larger studies. The next two chapters present tests of how well the design described in this chapter achieves these goals. Chapter 6 summarizes what the results of these tests reveal about how easily TESS can be used with various stimuli, how reliably it tracks text movements and how well it scales. Chapter 7 speculates on how the design described in this chapter could be improved or extended.

Chapter 4

Evaluating TESS for Correctness

This chapter describes experiments that evaluate the accuracy of TESS and their results. TESS codes user actions from screen recordings using a pipeline with several sub-steps, as illustrated in Figure 3.1. Each evaluation test examines one or two of these sub-steps. Taken together, these tests demonstrate that TESS works well in many cases. The points of failure are predictable and occur in less common reading environments.

The experiments described here test the internal workings of TESS and concentrate on parts of the pipeline depicted in Figure 3.1. Not all of the errors that are detected in these sub-steps affect the usefulness of TESS's final output. Chapter 5 describes reading experiments that use TESS and Chapter 6 presents an analysis of both this chapter and Chapter 5 that examines what the results presented here reveal about TESS's usefulness for reading experiments in general.

4.1 Finding Transitions

The finding transitions step was tested by comparing the times TESS chose to segment a single 10 second video segment to those chosen by humans comparing the same frames. Three human volunteers were shown a pair of frames, f_1 and f_2, that differed by 10 seconds and asked to mark each pair they observed as either the same or different. If the frames were determined to be different, then a frame, f_3, halfway between f_1 and f_2 was extracted and the pairs (f_1, f_3) and (f_3, f_2) were added to a queue of frames to be shown to the participant. If the two frames were determined to be the same then no further frames were added to the queue. This process was repeated on frame pairs in the queue until the time difference between each pair of different frames was 0.1 seconds. The pairs of different frames that

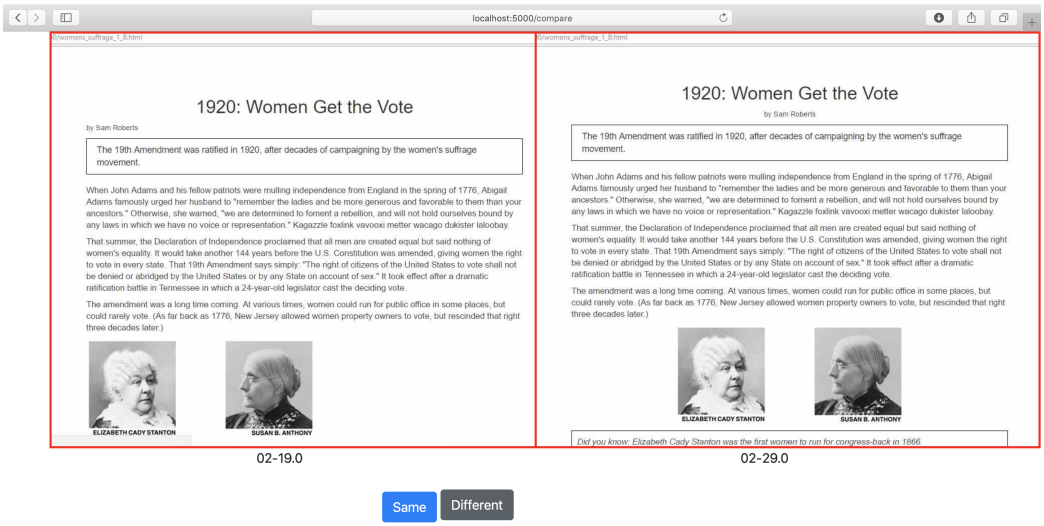


Figure 4.1: A screenshot of the evaluator's view during the transition finding test.

differed by only 0.1 seconds were recorded as the transitions found by the human subject. This experiment mimics the binary search step TESS uses to find transitions when T is 10 seconds and t is 0.1 seconds. A screenshot of the interface used by human volunteers is shown in Figure 4.1.

The human subjects found 4-6 transitions in the video segment, as shown in Figure 4.2. TESS marked 13 transitions for the same 10 second clip. All but one of the transitions found by the human volunteers was marked by TESS, and that transition was only identified by one of the human participants. TESS found a transition that was off by t from this transition. It is preferable that TESS find too many transitions than that it miss a few since the negative cost of processing extra transitions is computing time, while missed transitions can result in missing user actions and generating false data. The impact on run time of finding extra transitions is discussed in Section 4.8.

The human participants took 6 minutes on average to find the transitions in this 10 second clip while TESS took 32 seconds. A more complete picture of how long TESS takes to find transitions was computed by timing how long it took TESS to find transitions in 388 video clips ranging in length from 4 seconds to 25 minutes. Finding the transitions took TESS 15 seconds to 19 minutes per clip. TESS found 7-817 transitions in the clips,

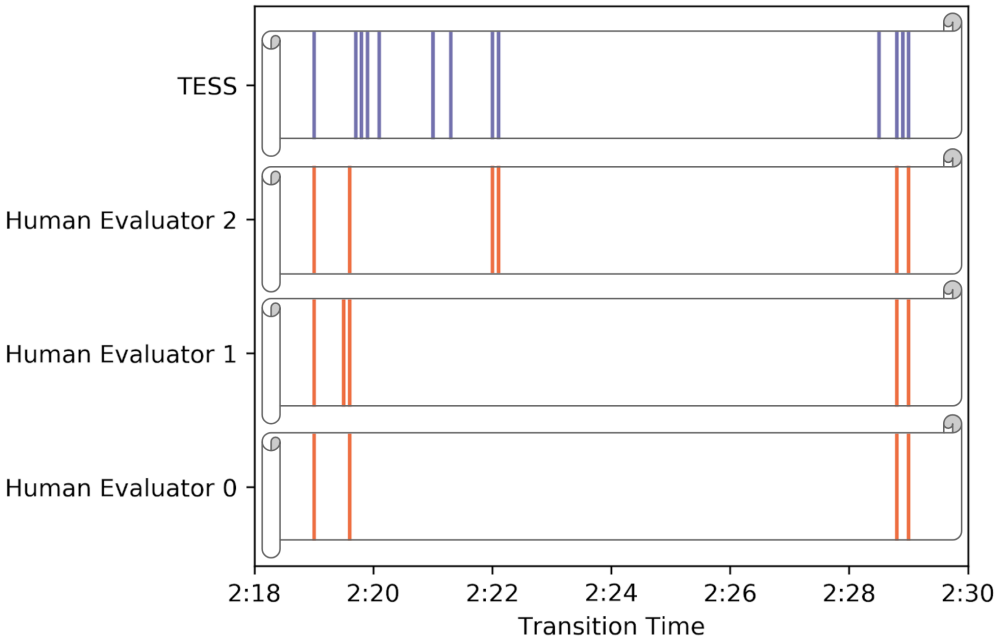


Figure 4.2: The transitions found by TESS and the human evaluators in the video clip. The times at

and the number of transitions found had a stronger relationship with how long the clip took to compute than its length based on a linear fit model. The model found that TESS took 21 seconds regardless of the number of transitions. Each additional transition took 1.3 seconds to compute. TESS's overall runtime is discussed in Section 4.8.

The design for this evaluation test was based on pilot testing I performed on myself. In early tests, I tried finding the transitions by watching screen recordings in a video player, but found it very hard to mark exact times of 'transitions' by scrubbing the video. This was particularly true when the change was continuous, as is the case during a scrolling action. Comparing still images was much easier, so I asked the human coders to mark images as either the same or different during a binary search. Since many of the volunteers for this evaluation test also volunteered for the test in Section 4.4, this test was purposely kept short and only involved one video clip. The video clip used for this study was chosen due to the relatively large variety of transitions that occurred during the clip. In this clip a webpage is

loaded, causing several HTML elements to move around, and the user starts scrolling near the end of the clip. Thus this video clip contains both localized and full screen transitions.

Based on these results, TESS segments screen recordings approximately twice as often as necessary. There is some evidence that it sometimes misses transitions, but since the missed transition was not agreed on by all the human coders, more evaluation tests are needed. It is better that TESS segment videos too much than too little, but it would be good to reduce the number of redundant time windows since TESS's running time is directory tied to the number of transitions it finds. This relationship is discussed in more detail in Section 4.8. Even with a slower than optimal runtime, TESS's video segmentation is 11 times faster than human coders.

4.2 Preparing Images

TESS can be setup to work with any OCR engine that outputs hOCR files. All OCR engines perform better under some circumstances than others. This section investigates the value one method of altering frame images to improve the results of the most popular open source OCR engine, Tesseract. Tesseract works best when all text in an image is at least 20 pixels tall [217]. Thus, frames that showed text that was only 10 pixels tall were run through TESS with and without re-sizing. The results indicate that Tesseract produces much more accurate text labels when the frames are resized, but TESS's post-text-correction-step results are equally accurate in both conditions. This test only looked at a few frames so it would be foolish to conclude that preprocessing images is never worth it, but the results show that increasing the accuracy of the uncorrected OCR results does not always increase the accuracy of the final output.

Four frames that contained 10 pixel tall text were used in this test. In the condition with prepossessing, each frame was resized to twice its initial dimensions without interpolation. The frames were chosen because they each posed a hurdle for TESS. Three of the frames included an open sidebar that covered part of the text. The fourth frame contained three

Frame	Before Text Correction		After Text Correction	
	No Prepossessing	Prepossessing	No Prepossessing	Prepossessing
1	44.5%	7.5%	0.0%	0.4%
2	33.3%	3.1%	0.0%	0.0%
3	37.1%	7.3%	1.6%	0.0%
4	32.8%	4.4%	2.6%	0.5%

Table 4.1: The word error rate for the four frames with and without prepossessing and before and after the text correction steps.

different colors of text (black on white, black on yellow and white on dark blue) and a small pop-up that covered two words. Two of the frames that included the sidebar also contained highlighted text. Section 4.5 discusses the effect of color on TESS's output. Other groups using Tesseract have noted that it works best for black text on a white background [217].

A manual inspection of the OCR output for each frame found that the word error rate before the correction step was 33-45% without resizing and 3-8% if the image was resized. The output for each frame was manually inspected again after the text label correction step. The word error rate was 0-3% for the non-resized frames and 0-0.5% for the resized frames. The exact error rates for all frames are shown in Table 4.1. Individual frames had different word error rates in each condition. In frame 1, text in the sidebar was incorrectly labeled as part of the document when the frame was resized, but not when it was analyzed at the original size. Frames 3 and 4 both had more errors after correction if they were not resized first. On average there were more errors if the frames were not preprocessed both before and after correction, but the reduction in the error rate was not as large after the text correction step. A full investigation of the value of prepossessing would need to use a larger sample.

This test shows that improving the initial OCR results does not always lead to a proportional improvement in the results after the text label correction step. The test performed here only looked at four frames, so this is not definitive evidence of the value of resizing images, but it does show that adding preprocessing and improving the OCR engine do not

always lead to more accurate final text labels. Preprocessing can be seen as a proxy for investing in a better OCR engine. Resizing the frames in this test reduced the initial word error rate by 85% on average, but for half the frames the final word error rate was as good or better without resizing. Thus researchers who use TESS may not need to purchase a state of the art OCR engine.

4.3 Document Matching Tests

The document matching part of the OCR correction step was tested by taking screen shots of 80 Wikipedia articles and checking how many of these screen shots TESS successfully paired with the text of the correct article. The pages were selected by starting with the Wikipedia page on cats and randomly following links. At each page, the text of the page was copied into a text file which was labeled with the name of the page. Then the page was scrolled down a random amount and a screen shot was taken of the page. Since this test only looks at document matching, no attention was paid to line breaks in the text files.

For each text file and hOCR file pair, the document similarity was calculated. In all cases, the text file with the largest document similarity score to the OCR output for each screen shot was the correct article. To observe how robust the document similarity score is, the ratio between the similarity score with the correct document and the largest similarity score with another document was also calculated. A histogram of these ratios is shown in Figure 4.3.

The results show that the similarity metric consistently matches the hOCR files to the correct document. The document similarity to incorrect documents was never more than 85.3% the similarity to the correct document. For all but one hOCR file, the largest similarity score to an incorrect document was less than 80% that of the similarity score to the correct document. The one outlier was a Wikipedia article that was only one paragraph in length. This article described the Battle of Barbalissos between the Sassanid and Roman Empires, and its second closest match was to the article on the Sassanid Empire. The re-

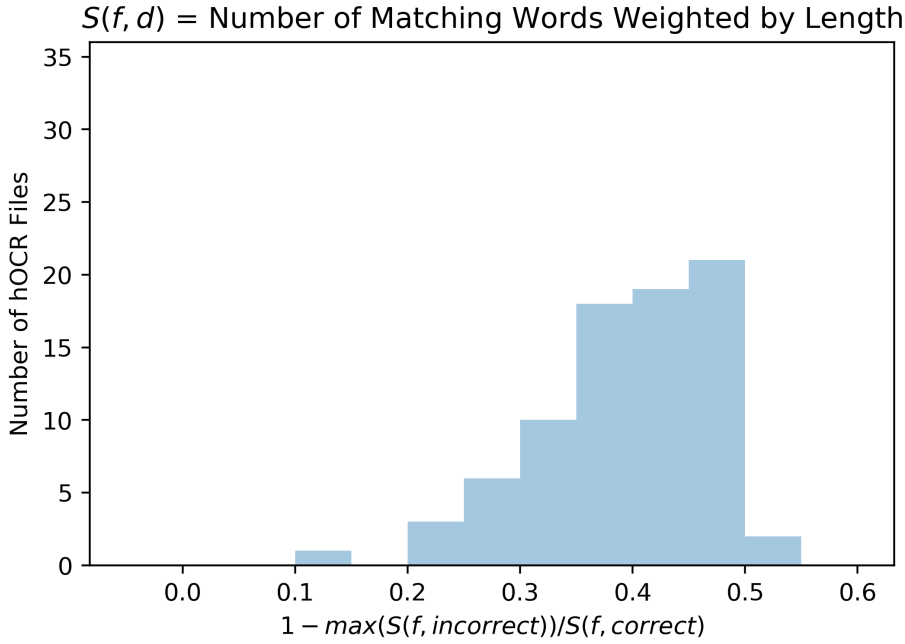


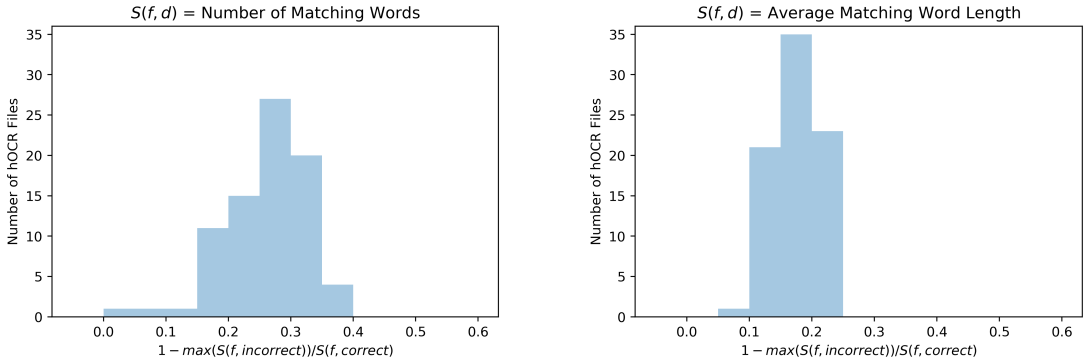
Figure 4.3: One minus the ratio between the similarity score of frame and the correct document and the largest similarity score between that frame and an incorrect document. The ratios are subtracted from one so that the closest values will be near zero.

sults of this test show that the similarity score is robust for a reasonably large corpus, even if the documents in the corpus are short and discuss similar subjects.

As described in Section 3.2.1, the document similarity function TESS uses is

$$S(f,d) = \sum_{token \in (f \cap d)} length(token)$$

To test whether another similarity score would work better, the test described above was run with a similarity score that does not weight the words by their lengths and a similarity score that uses the average word length and thus does not take the number of matched words into account. These alternative similarity functions also correctly matched all the hOCR documents, but did not have as wide a buffer between the similarity score for the correct document and the largest similarity score to an incorrect document as shown in Figure 4.4. When word length was not taken into account, one hOCR file had a similarity



(a) Word Count

(b) Average Length of Words

Figure 4.4: One minus the ratio between the similarity of an hOCR file with the correct document and most similar incorrect document under different similarity metrics. The ratios are subtracted from one so that the closest values will be near zero.

to an incorrect document that was over 99% its similarity to the correct document. Thirteen of the 80 hOCR files had similarity scores to incorrect matches that were over 80% their similarity score to the correct document. When the number of words matched was not taken into account, 57 of the 80 hOCR files had similarity scores to incorrect matches that were over 80% their similarity score to the correct document. There is probably a better document similarity function than the one used by TESS, but it is not counting the number of shared words or taking their average length.

This test did not look at screen shots that only contain a few words or examine languages other than English. It is possible that the document similarity function would become less reliable under these conditions. For the moment, TESS uses a single similarity function that works well for screen recordings of large articles in English. As more researchers incorporate TESS into their studies, it may become necessary to swap out the similarity metric depending on the text language or the amount of text on the screen.

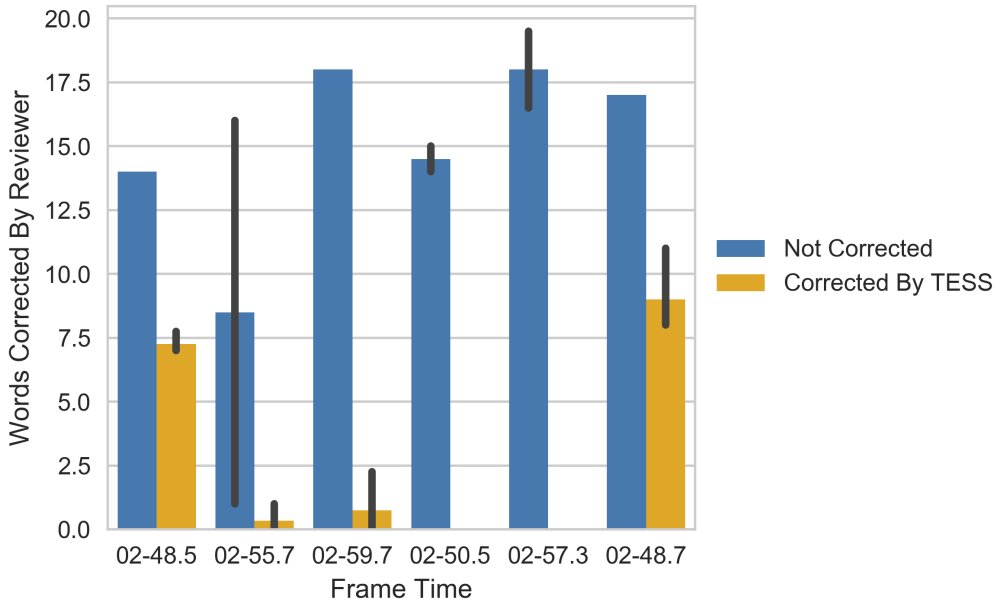


Figure 4.5: The number of words human reviewers marked incorrect in each frame before and after correction by TESS.

4.4 Line and Word Correction

The validity of the correction step after a frame has been matched to a document was assessed by five human reviewers who examined OCR results before and after correction. Each reviewer was shown six frames from a single reading experiment, which were randomized to either be the uncorrected output of the OCR engine, or the output after correction. The output was shown in TESS's interactive editor (described in Section 3.6). Reviewers edited each frame until the text was correct as they saw it.

Both the uncorrected and corrected version of each frame was seen by at least one reviewer. The reviewers did not always find that the frames corrected by TESS were error free, but they did consistently find fewer errors than the frame's uncorrected counterpart. Figure 4.5 shows how many words were marked incorrect in each condition, and the standard deviation for frames that were viewed by more than one reviewer. In the frames for times 2:50.5 and 2:57.3 no errors were found in the post correction version.

The average word error rate for each frame before correction according to the human reviewers was 3-6.5%. Fifteen percent of the edits reviewers made only involved changing punctuation or the case of a character. After the text correction step the error rate dropped to 0-3.3%. All the edits reviewers made to corrected frames added text that had been deleted by TESS. In most cases, this error came from an entire line being deleted from the top of the frame. The deleted words were all partially covered by the top of the page.

As mentioned in Section 3.2.2, lines that did not get matched via correctly identified words and whose weighted distance to the lines in the text file is over a threshold are deleted by TESS's text correction step. The partially covered lines that were incorrectly deleted from the top of the frame in this test had no words recognized correctly by the OCR engine and did not have a sufficiently small edit distance to the correct line in the text file to be matched.

Deleting lines that were partially hidden by the top of the frame is probably not a big deal since participants are unlikely to read lines without making them fully visible. However, this may be an indication of how TESS would fail to match lines if the OCR engine was less reliable. In cases where a strong OCR engine is not available, as is often the case for less studied alphabets, TESS may need to use a lower threshold for matching with the distance metric. Finding the right threshold is a difficult dance, since too low a threshold will cause menus and other extraneous text to be matched to lines in the correction documents.

This test evaluated TESS's line and word matching together. The results show that both correction steps occasionally make errors, but line matching has a larger impact. Overall, the correction step consistently lowers the error rate and the remaining errors are predominantly in partially covered words which users are unlikely to look at. This test used frames that displayed black text on a white background. The error rate after correction is often lower when a large range of highlight colors are used as discussed in Section 4.5, but the test described here evaluates TESS's performance in environments that use the most popu-

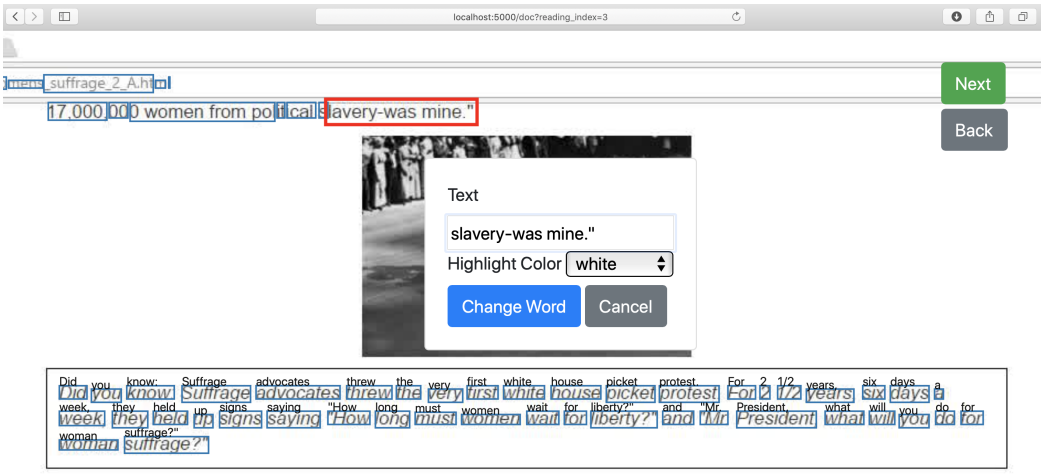


Figure 4.6: A reviewer marking text incorrectly deleted by TESS.

lar text color.

4.5 Color Detection Tests

Three tests were used to monitor TESS's word color detection accuracy. The first compared TESS's color detection to another highlight tracking application. The highlights tracked by the application had black text on a yellow background. The text that was not highlighted was black text on a white background. Both of these color pairings consist of a dark and a light color. Thus TESS can use the faster threshold method to group pixels into a text color and a background color.

To test TESS's ability to correctly detect a larger set of colors, a second test was run by randomly coloring the text and background of each word in a webpage from a set of 25 color pairs. Screen-shots were taken of the webpage and TESS's color detection was compared to the colors assigned for each word. This test was run using both the clustering and thresholding methods of separating text and background pixels so the differences in time and accuracy between the two methods could be compared.

The third test measured TESS's accuracy on a data set which contained more colors than the first test, but unlike the second test used popular highlight colors that were easy to

read. This test was manually evaluated using the interactive editor described in Section 3.6. The results of the three tests indicate that TESS's text labeling and color detection accuracy are dependent on the colors used. In general, the clustering method of separating pixels is more accurate but slower and TESS works best for dark text on a light background. Having a small number of possible highlight colors that are not close to each other also helps.

4.5.1 Single Highlight Color Test

TESS's accuracy at detecting the color of words in a classical example (one default color, one highlight color) was tested by comparing TESS's results to the highlights saved by the Hypothes.is JavaScript plug-in [95]. For this test, 107 participants read and made highlights while their screen was recorded. Saved highlights were stored by the Hypothes.is JavaScript plug-in during the screen recording. Since Hypothes.is only contained a list of the final saved highlights (not highlights that were made then deleted during the session), TESS only examined the final frame in each screen recording. Participants were not encouraged to make highlights after the screen recording stopped, but they continued to have access to the screen while they completed a post-test. There is some evidence that some participants may have saved new highlights after the screen recording stopped. The words detected by TESS in each final frame were marked according to whether they were saved as part of a highlight on Hypothes.is. TESS's color detection step was then run on each frame to find any differences between TESS's results and Hypothes.is's.

Participants saved highlights on hypothes.is in 60 of the 107 sessions. In total, hypothes.is recorded 250 highlighted snippets made up of 1,252 highlighted words. This left 26,601 non-highlighted words across the 107 frames. TESS agreed with Hypothes.is about 26,590 of the 26,601 (99.96%) of the non-highlighted words, and 892 of the 1,252 (71%) highlighted words. The differences came from 15 of the 107 participants. A manual examination revealed that in one instance, TESS classified a word as a highlight due to the darker colored address bar in the surrounding area. In five instances, words were partially

Figure 4.7: A snippet of the website in which each word is randomly colored from one of 25 color pairs.

highlighted and TESS classified them as not highlighted. The remaining 365 discrepancies appear to come from 13 text snippets recorded by Hypothes.is that are not highlighted in the participant's video and 3 text snippets that are highlighted in the video but not recorded by Hypothes.is.

The 13 snippets that are not in the video likely result from participants making highlights during the post test (which was not part of the screen recording examined by TESS). The 3 snippets that are highlighted in the video but not recorded may be due to the save button not being properly pressed while using the Hypothes.is tool (experimenters were able to replicate this bug in the Hypothes.is tool). The results of this investigation show that TESS is able to recognize the colors of words very accurately when the entire word is highlighted and unobstructed.

4.5.2 Large Color Range Test

The test with the Hypothes.is tool discussed in Section 4.5.1 looks at TESS's performance when there is only one default color pair and one highlight color pair, both of which consist of a dark and a light color. To test TESS's performance more generally, a second test was devised in which the text and background of each word in an image were colored randomly. The colors assigned to each word were saved when the HTML file was generated so TESS's results could be compared to the assigned values. An image of some of the colored words is shown in Figure 4.7.

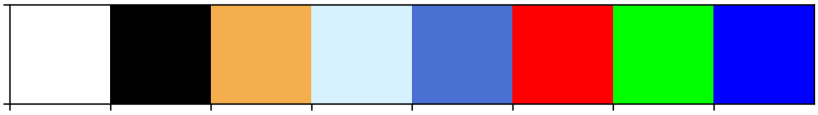


Figure 4.8: The eight colors used for the text and background of each word.

The images for this test were generated by randomly coloring the text and background for each word from a list of 8 colors. Since each word uses two colors (one for the foreground and one for the background), the list of possible colors for each word contains more than 8 possibilities. Pairs in which the text and background are the same color were excluded since that text would be illegible. TESS is not setup to tell the difference between a color pair in which the text is white and and the background is black, and the reversed color pair where the text is black and the background is white. To ensure that only one of these pairs exists in the set of possible colors, only pairs in which the average brightness of the text color is darker than the average brightness of the background were used. Three of the colors (red, green and blue) had the same average brightness, so the filter also excluded the 3 pairs made up of these colors. After filtering, there were 25 color pairs that could be assigned to a word.

Five images, each containing 199-310 words, were generated for this test. In total, the images contained 1,382 words. Each of the 25 color pairs occurred between 38 and 74 times across the five images. The frequent color switching in the images confused the OCR engine. As a result, even after correction the text labels had many errors. A manual correction was used to fix the labels of the detected words for each image using the interactive editing tool described in Section 3.6. This manual correction revealed that 59% of the word blocks were assigned the the wrong labels. Of the 1,382 words, 23% were not detected by the OCR engine. TESS has no methods for adding undetected words, so these words were not analyzed. Of the 1,069 words that were detected, 680 (64%) were not detected as their own word block but were instead grouped with at least one other word. TESS's color detection only works if all the words in a block are colored using the same

color pair. When there are only a few possible color pairs, this can happen even if more than one word is in the block, but when there are 25 color pairs that are assigned randomly, the probability that a pair of consecutive words will be different colors is .96.

The frequency with which words went undetected or were grouped into blocks with other words was not the same for all colors. The pair of light blue text on a white background went undetected in 98% of occurrences. This is the color pair used for '"we' (after 'she warned,') in Figure 4.7. The other color pairs were missed by the OCR engine 10-31% of the time. Notably, the light blue text on a white background color pair had the smallest brightness difference between the colors in the pair. Thirteen of the 25 color pairs were grouped with other words in 77-100% of their occurrences. The other half of the color pairs were grouped with other words 28-50% of the time. As shown in Figure 4.9, all the color pairs in which the average brightness of the colors differed by less than 40% (or $\frac{100}{256}$) of the brightness scale were in the first group. If a color pair had more contrast, words colored with that color were less likely to be grouped with another word.

After the manual correction, there were 389 words which were detected by the OCR engine and were alone in their word block. Twenty-three of the original 25 color pairs were represented by these words. The thresholding method was able to correctly label the color pair in 66% of cases, while the clustering method correctly labeled the color pair in 81% of cases. The increased performance of the clustering method came from three color pairs: the orange-yellow color on top of white or light blue, and black text on the medium blue background. The orange-yellow on light blue color pair is used to color 'laloobay.' in Figure 4.7. The clustering method had an error rate of 3-4% when the text was the orange-yellow color on top of white or light blue, and 60% for black text on the medium blue background. The thresholding method failed 100% of the time on all three of these color pairs since both colors were on the same side of the threshold.

There were four color pairs where thresholding out performed clustering, but the difference in performance was not as extreme as the cases where clustering did better. For

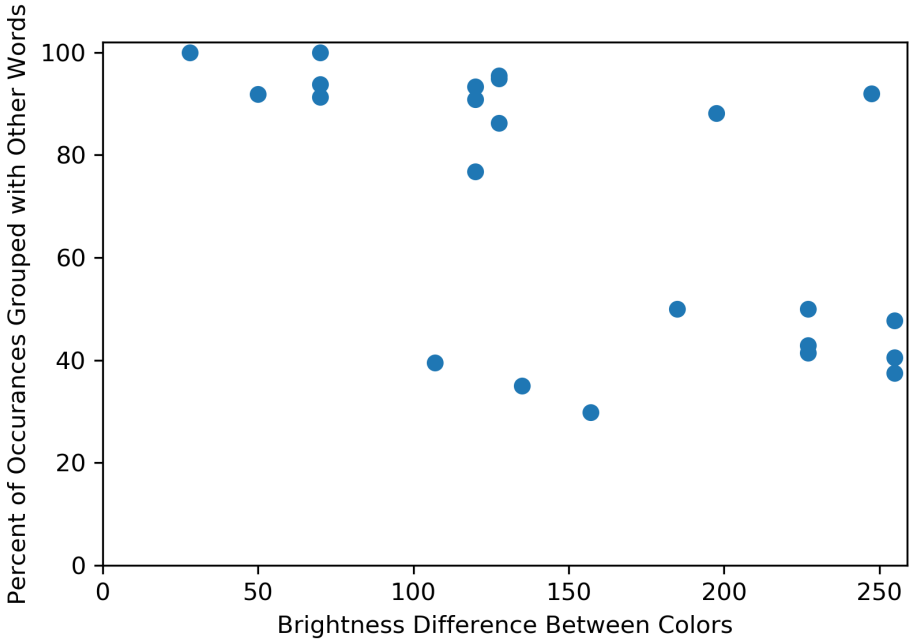


Figure 4.9: The probability that words colored using a given color pair will be grouped with other words by the OCR engine as a function of the average brightness difference between the two colors.

dark blue text on a white background or red text on a light blue background, the clustering method made a mistake in 1 and 3 more instances respectively than the thresholding method (out of 30 and 36 total instances respectively). The clustering failed in half the cases where the text was medium blue and the background was the orange-yellow color. In all cases where the text was red and the background was the orange-yellow color the clustering method failed. The thresholding method made no errors in either of these cases.

An analysis of the words with red text on a light blue background and dark blue text on a white background found that clustering failed on these colors because the background color did not extend past the word's bounding box. When analyzing the area of an image that displays a word, TESS includes a small border around each word. These extra pixels are usually the background color since highlighting and selecting tools usually color a small border around each word in addition to the word itself. Words that only have one or two letters don't contain many background pixels in the bounding box that encapsulates the

Figure 4.10: How an image snippet of a red and orange-yellow word with a white border is partitioned by the clustering and thresholding approaches.

letters of the word, so adding this border allows TESS to see necessary pixels for finding the background color.

The span elements used to color each word in the images used for this experiment did not extend the background color past the word's bounding box. Thus, the border usually picked up white pixels, which were sometimes recognized was the second cluster in the clustering approach if the text and background colors were sufficiently close to each other. These white border pixels were always included in the lighter pixel group when pixels were divided using the thresholding approach. Since the darker color was always used to color the text in this experiment, this was always the background color. The median background color usually remained the same in the thresholding approach as the background pixels were more numerous than the white border pixels when the word contained at least three letters. An example of a red and orange-yellow word with a white border divided by the clustering and thresholding approaches is shown in Figure 4.10.

There were seven color pairs in which both the thresholding and clustering methods assigned incorrect colors for 100% of word occurrences. An examination of these color pairs revealed that thresholding usually failed because the colors in the pair were on the same side of the threshold. For two of the seven colors, clustering partitioned the image correctly, but measured the distance from the median values in each partition to be closer to an incorrect color pair than to the correct color pair. For the other five color pairs, the clustering method chose the white border as one of the clusters as it did for the red and

orange-yellow color pair shown in Figure 4.10.

Based on these results, the clustering method was re-run without the border pixels for each word. Removing the white borders increased the accuracy of the clustering method from 81% to 85%. Most of this improvement came from the color pairs which had been incorrectly categorized by the clustering method in 100% of cases previously. Once the white borders were removed, only 2 color pairs were consistently miss-categorized by the clustering method.

The difference function used to match detected colors to values on the list of possible color pairs was also updated to weight difference in green pixels as five times more important than differences in blue pixels. This altered difference function was meant to reflect how difference is perceived by the human eye, which is more sensitive to gradations in green light than blue light. Using the altered difference function without border pixels brought the accuracy up to 89%. However, an examination of the results showed that this result came from a change in how words that had green text on either an orange-yellow or light blue background were categorized. The altered difference function categorized the background as light blue in both cases, while the original difference function categorized the green text as orange-yellow when the background was light blue. There were more instances of green text on a light blue background in the sample, so the altered difference function had a lower overall error rate in this example. Both difference functions have trouble distinguishing among these colors when the pixel values are slightly off. Human observers do not have trouble categorizing the median colors found by the clustering method, as shown in Figure 4.11. A better difference function that is closer to the algorithm used by the human eye would probably categorize these color pairs correctly.

Finding word colors using the thresholding method took 2 seconds for all 5 images. Using the clustering method this same process took 32 seconds. The experiment shown here demonstrates that the clustering method yields better results if a wide range of color pairs are used and no threshold divides all of them. However, the range of colors in this

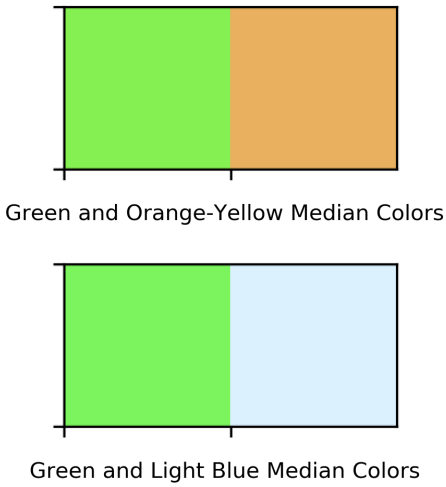


Figure 4.11: The median color values for each cluster in a green and orange-yellow word and a green and light blue word. Tess's difference function categorized the orange-yellow in the top image as light blue and the green in the bottom image as orange-yellow.

experiment is much wider than in most reading environments. Studies of human readers have found that many struggle to read low contrast text [28, 75, 83] and prefer dark text on a light background [175], so most reading experiments use high contrast color combinations. The OCR results in this experiment demonstrate that researchers using a large number of color pairs with low contrast may want to invest in a better OCR engine.

4.5.3 Four Color Test

The experiment in Section 4.5.2 shows that TESS's text labeling and to a lesser extent color categorization are unreliable when the set of colors used grows sufficiently large. However, the images in the Section 4.5.2 experiment are not representative of most reading experiments, and some of the color pairs are difficult for humans to read. The experiments in Sections 4.5.1 and 4.4 show that TESS performs much better when the text is easy to read, but don't investigate TESS's color detection accuracy in authentic reading environments with more than two colors. The test described in this subsection measures TESS's performance in a real-world reading environment with four colors.

To measure TESS's error rate within a small set of easy to read color pairs, the interac-

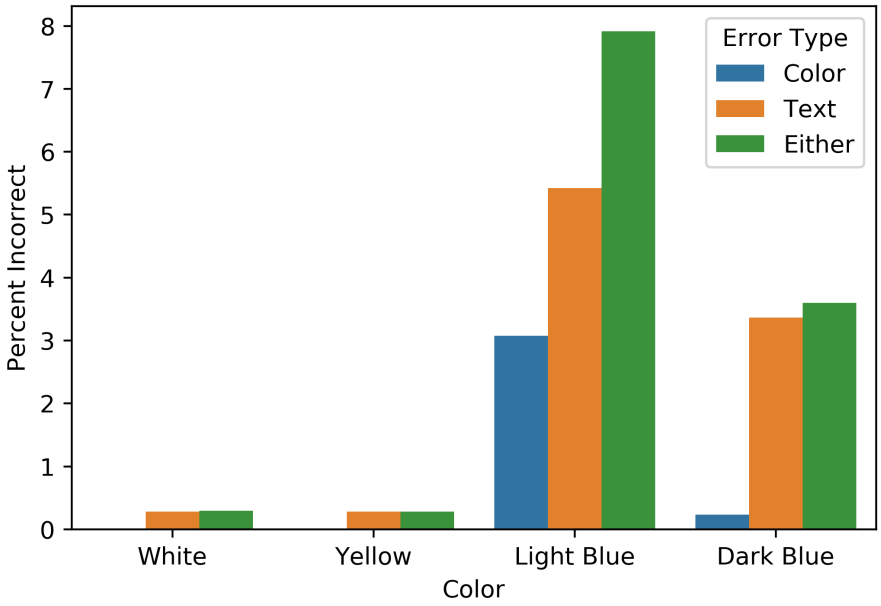


Figure 4.12: The error rate and type of error (color detection or labeling) for different colors. All colors refer to the background color. When the background was dark blue the text was white.

tive tool was used to correct errors in the detected highlights for 3 video clips. Each clip contained 5-10 highlights saved using hypothes.is. In addition to yellow saved highlights, TESS detected when text turned dark blue or the background turned light blue due to selection actions that were not saved by hypothes.is. The colors and labels for each word after corrections were manually made with the interactive editor were compared to the output before manual editing. Among words that were labeled white or yellow, 0.3% had an error in either their label or color. The error rate for words originally labeled dark blue was 3.6%, and for words originally labeled light blue it was 7.9%. Figure 4.12 shows the error rate for each color. Errors in which the wrong color was assigned to a word block are marked as 'color' errors. Errors in which the wrong label was assigned to a word block are marked as 'text' errors.

Most of the errors for dark blue words (which was white text on a dark blue background) came from incorrectly labeled text. Tesseract does not work as well for light text on a dark

background as dark text on a light background [217]. Errors from the initial OCR output were often only partially corrected, particularly in cases where word separation was not properly recognized. In 53% of the 87 cases that a dark blue word was mislabeled, the 'word' was a group of at least two words. When a dark blue 'word' contained multiple words, the correction step assigned at least one of the correct word labels to that 'word' in 46% of cases. Among dark blue 'words' that were correctly labeled, only 0.9% contained multiple words.

Words labeled light blue were the most likely to contain errors. These errors came from a variety of sources. In 18% of the light blue errors, a text block that was not part of the stimuli was mislabeled as part of the article. These text blocks were usually marked light blue because that was the closest color to the colors of the address bar and menu. Light blue was also the closest color to the white default color (97% of words across all frames were white). Smaller text, bold text and text near images or the address bar was more likely to be incorrectly marked light blue due to noisy pixels or bounding boxes that overlapped other objects like pictures.

Errors in highlight detection could sometimes be spotted by looking for instances where a highlight appeared to be created and deleted very quickly. As the analysis above shows, a manual check for highlighting errors should focus on longer word blocks, words near pictures, and words detected near the top of the page. In future work it may be possible to automatically correct some of these errors.

4.5.4 Color Detection Test Summary

While the evaluation test in Section 4.5.2 demonstrates that TESS's color detection step makes errors, the tests in Sections 4.5.1 and 4.5.3 demonstrates that its outputs are usually accurate when the set of possible highlight colors is reasonable. When words are colored randomly, the OCR engine is likely to miss word borders. However, TESS is designed to be used in reading experiments and readers do not highlight randomly. Most highlights

cover phrases, not single words, and most readers highlight less than half the words they see. Highlight colors tend to have high contrast since that makes them easier for humans to read. Fortunately, OCR engines also find high contrast text easier to read.

With that being said, TESS's inability to detect more than one color in a word block limits its usefulness. It is not unheard of for readers to partially highlight words, particularly younger readers have developed all the fine motor skills and tricks adult readers employ to select entire words. For example, adults familiar with computers often learn that double clicking on a word will select it in most applications, but this knowledge is not rarely taught in schools. A version of TESS that could detect multiple highlight colors within a word block would be useful both for cases when the OCR engine fails to find a word break and for cases when participants partially select words.

4.6 Action Coding

Once TESS has analyzed each frame, it detects user actions by examining differences between adjacent time windows as described in Section 3.4. Calculating these differences is straight forward, but errors occur as the result of errors in TESS's analysis of individual frames. Not all errors in frame analysis result in the same magnitude of mistakes in action coding. For example, if TESS's measurement of a line's position is off, the detected scrolling might still be correct since the scrolling detection step examines more than one line. If TESS detects the incorrect color for a word, there will be a mistake in TESS's highlighting report, but if the correct color is detected in the next frame and the time window is small, then this will stand out as a probable mistake and researchers using TESS can filter it out. This section examines the errors present in TESS's action coding output.

TESS is built to be compatible with all reading environments, so it does not make assumptions as to how users scroll or which color transitions are possible. Thus by using screen recordings from an environment where jumping and highlighting were restricted, errors in TESS's output can be easily detected. The evaluation tests in this section use

data collected while readers read a document without internal links (so jumping to random locations within the page was impossible) and where only some color transitions were possible. Using data from this restricted environment makes it easier to find mistakes. Scrolling detection was tested by examining a few screen recordings in detail and checking that jumping behavior was absent from the full set of results. Highlighting detection was tested by comparing highlighting reports before and after using the interactive editor to fix mistakes. The results indicate that scrolling detection is robust and reliable. Highlighting detection is more prone to errors, but most of the errors mistake the length of time a word was highlighted, not which words were highlighted at all. Statistics on what words users highlighted were almost always correct and statistics on how long text was selected were usually off by less than 1%.

4.6.1 Scrolling

One hundred and ninety one middle school students were asked to read an article with no links to test TESS's scrolling detection. Participants could scroll by using a keyboard, a mouse or a touchscreen display. In theory a session could contain a lot of scrolling because a reader jumped from the top to the bottom of the screen many times. In practice, most readers only go through an article once or twice. In articles with lots of links, a reader may take a non-linear approach to a first reading of the text. In articles without links like the one used here, the environment discourages readers from reading out of order and most readers move linearly toward the end of the article. Thus, just as the walker with the most steps will have the shortest stride, readers whose scrolls are large in terms of the number of pixels moved should have fewer instances of 'scrolling actions' detected.

To check whether students who scrolled more, scrolled shorter lengths than their peers, the number of downward scrolls TESS detected in each recording was plotted against the average number of pixels moved by each scroll. If the student zoomed in, their scrolling while zoomed in was excluded from this analysis, since it was easy to accidentally scroll

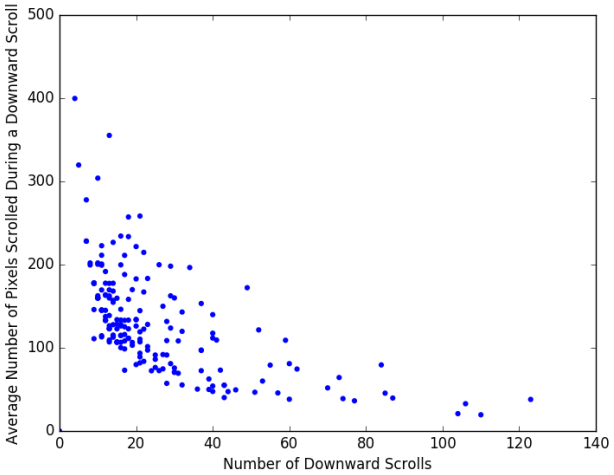
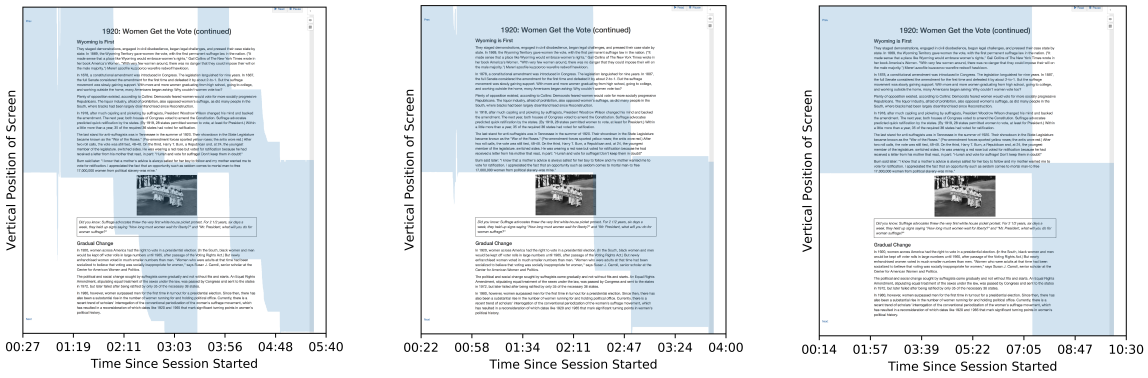


Figure 4.13: The average pixels scrolled plotted against the total number of downward scrolls.



(a) Maximum Scroller (b) Median Scroller (c) Minimum Scroller

Figure 4.14: Visualizations of scrolling from different points in the distribution.

while attempting to zoom when using the touchscreen. As shown in Figure 4.13, there was a relationship between the number of times TESS detected a reader scrolling and the average length of the detected scrolls, with those who scrolled less often making longer scrolls.

Statistics occasionally obfuscate more than they reveal, so three participants' scrolling behaviors were manually reviewed. The three students were those with the most 'scrolling actions', the median number of scrolls and the fewest 'scrolling actions' according to TESS. Figure 4.14 shows visualizations of each of these participants scrolling. The frequent move-

ment of the blue band in Figure 4.14a shows that the student scrolled quite a bit, while the stability of the blue band in Figure 4.14c shows that the student only scrolled in one short burst a little over 7 minutes into the session. This test demonstrates that the statistics plotted in Figure 4.13 are not a fluke, and TESS's conception of heavy and light scrolling matches the common understanding. The study of scrolling behavior discussed in Section 5.1 used these results to demonstrate that the scrolling data is reliable.

4.6.2 Highlighting

Correctly identifying highlighting actions requires TESS to correctly identify the color and assign word labels correctly regardless of word color. The experiments in Section 4.5 show that TESS's accuracy varies a lot depending on text color. To see how errors in individual frames affect the aggregated highlighting results, the aggregated highlighting actions for the data set with four highlight colors described in Section 4.5.3 before and after manual correction were compared.

The manual correction was performed by finding the transitions where words changed color with the highlighting visualization described in Section 3.5.3, and reviewing frames on either side of the transition with the interactive editor described in Section 3.6. The environment used for this experiment only allowed one or two color transitions from each possible color pair. White words could only turn dark blue. Dark blue words could turn yellow or white. Yellow words could turn light blue or white, and light blue words could turn yellow. The manual correction concentrated on impossible color transitions and words that changed back and forth quickly.

The data set contained three videos, each showing the screen of someone reading and highlighting a document. In one video the reader viewed a document with 434 words, while in the other two videos the participant viewed documents with 674 words. Of the 1,782 words observed in the three videos, TESS flagged 260 as changing color at least once. Ten (4%) were found to not actually change color after the manual correction. The

corrected results showed that 254 words changed color across the three videos, 5 (2%) of which were not found by TESS. Most of these errors were caused by highlighted words being mislabeled.

Among the 250 words that TESS correctly flagged as having changed color, 43 (17%) were labeled the wrong color at least once in the video. In 28 of these cases, the discrepancy came from TESS marking the word as not highlighted during a period of time that it was highlighted. On average, these 28 words were incorrectly marked not highlighted for 15% of the time they were highlighted. The distribution has a long tail, half of the 28 words were incorrectly marked not highlighted for less than 1% of the time they were highlighted. Twelve words were incorrectly marked highlighted when they were not. On average, 19% of the time these 12 words were marked as highlighted by TESS was an error. There were 3 words that were sometimes incorrectly marked as highlighted and sometimes incorrectly marked as not highlighted. In all three of these cases, the word was correctly marked as dark blue, but the detection of when it turned dark blue and when it changed back to the default white color was off by about a second. There were no cases in which TESS correctly identified a word as highlighted, but marked the highlighted word the wrong color.

The total amount of time that words were marked the wrong color is shown in Figure 4.15. As discussed above, the distribution has a long tail and most color errors persisted for less than a second. In future work it may be possible to automatically tag suspicious highlights based on the length of time a highlight is detected.

4.6.3 Action Coding Summary

The results discussed above show that TESS's scrolling detection is very accurate and detected highlights are correct most of the time. Furthermore, researchers can use robust statistics such as medians to mitigate the propagation of errors. Filters such as ignoring scrolls and highlights that are too short can also be used to remove noisy readings. Examples of action statistics used in real world reading studies are presented in Chapter 5.

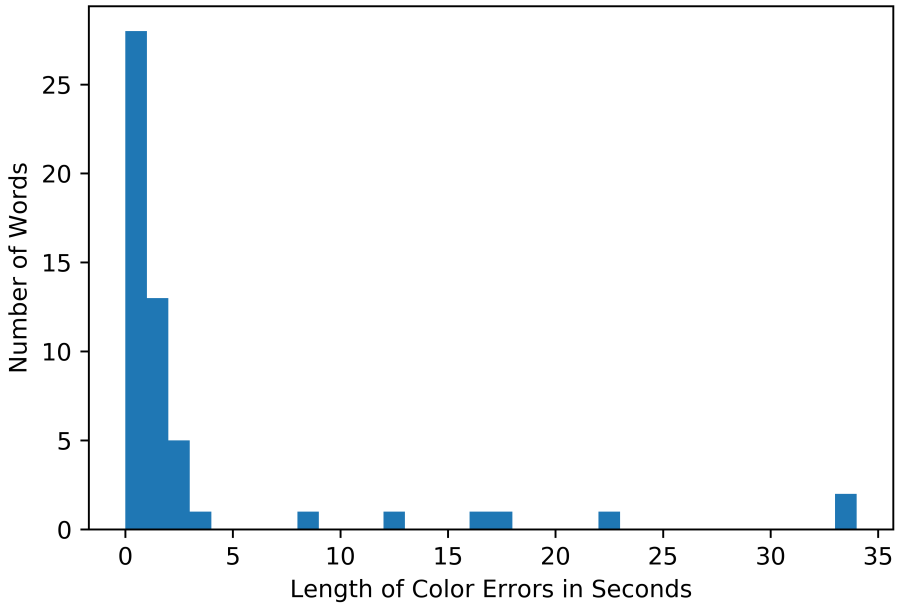


Figure 4.15: The amount of time that words were marked the wrong color across three videos ranging from 2.5 to 4.5 minutes in length each displaying 400 words or more

The evaluation tests presented in this section used the limitations of the reading environment to find errors quickly. While not all reading environments contain the same strict limitations, many documents encourage some types of actions and discourage others. Similar techniques can be used to quickly find errors in other reading environments by visualizing TESS's output and looking for behaviors that are strange given the reading task.

4.7 Visualization Tests

TESS is designed to assist researchers working on open questions around digital reading. The evaluation tests described in the previous sections demonstrate that TESS can track micro actions from screen recordings. However, many researchers are interested in larger scale questions that are composed of several micro actions like whether a reader scrolled quickly or 'skimmed' through the whole article [185, 207], the total amount of

time a reader dwells on a section [11], or how long it takes a reader to turn selected text into a saved highlight [154]. To code more subjective 'macro' actions, human coders can use visualizations of TESS's output. This section presents an experiment testing whether manually coding 'macro' actions from visualizations generated by TESS is more efficient than coding the same actions directly from the screen recording.

Action	Label	Description
Length Check	Scrolling 1	Scroll down and up very quickly
Jump to top	Scrolling 2	Scroll to the top very quickly outside of a length check
Select text without highlighting	Highlight 1	Select text in dark blue and then unselect it without turning it into a yellow highlight
Select text and highlight	Highlight 2	Turn dark blue text into yellow highlight
Delete highlight	Highlight 3	Delete yellow or light blue highlight so it becomes white again

Table 4.2: 'Macro' actions coded by visualization evaluators

Twelve subjects were asked to count 2 scrolling actions and 3 highlighting actions based on videos and visualizations from two screen recordings. The actions are listed in Table 4.2 and were purposely chosen to be somewhat ambiguous so the results would be applicable to complex coding. For example, selecting text may involve dragging the mouse across a text snippet, briefly unselecting the text and starting the drag again from a slightly different position. What constituted a single text selection for Highlight 1 was left up to the participant. Some of the actions, like Scrolling 1 [70] and Highlight 1 [213] were inspired by references to the action in the literature. Other actions were chosen because they appeared to be similar. The experiment was conducted via a website that contained written instructions for each subject. Once a subject agreed to take part in the study, they were given the web address for the study. The welcome page of the website asked each participant to make sure they had enough time to complete the study in a single sitting.

Participants were first shown instructions explaining what they would be asked to do in

Figure 4.16: The visualization view for video clip 2.

the study and a tutorial video that illustrated how the scrolling and highlighting visualizations should be interpreted along with examples of each 'macro' action. Each participant was asked to rate their understanding of the visualizations based on the video, answer a question designed to test their understanding of the scrolling visualization, and count the occurrences of 2 scrolling and 3 highlighting actions based on the video and visualizations given in the example. As mentioned in section 3.6, TESS often makes mistakes which are apparent in visualizations of its results. For this study, all the visualizations were cleaned up so the results would reflect the value of the visualizations once all of TESS's output was sanitized.

Once participants finished going through the instructions, they were shown four pages. On each page they were again asked to count the occurrences of the actions listed in Table 4.2 based on the video or visualizations on the page. The first page showed scrolling and highlighting visualizations for a second video clip. A screenshot of this page is shown in Figure 4.16. The second page showed the video representation of a third video clip. The

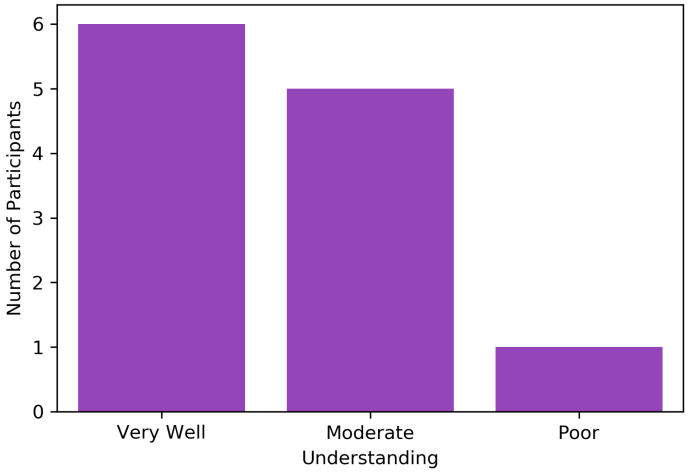


Figure 4.17: The self assessed understanding level of each participant after reading the instructions and watching the explanatory video.

third page showed the video representation of video clip 2 and the fourth page showed scrolling and highlighting visualizations for video clip 3. The first three participants were asked after the experiment if they realized pages 1 and 3 or pages 2 and 4 corresponded to the same clips, and all 3 said this was not apparent during the experiment. All participants were asked to give feedback through a text box on every page and at the end of the study. None of the participants mentioned noticing that the video clips repeated.

Eleven participants said they understood the visualizations 'Very Well' or 'Moderate'. The participant who said their understanding was 'Poor' did not feel that they were able to read the visualizations well enough to answer the counting questions, so their answers were excluded from the rest of the analysis. Notably, this participant was the only participant who had not completed a bachelors degree in a STEM (Science Technology Engineering and Math) discipline, which suggests that mathematical training is helpful when reading the visualizations. Figure 4.17 shows the self-reported understanding of each participant. Participants gave a wide range of answers to the question designed to assess understanding. In many cases it was clear that the participant was not able to understand what was being asked. For this reason, the answers to the understanding question were not used.

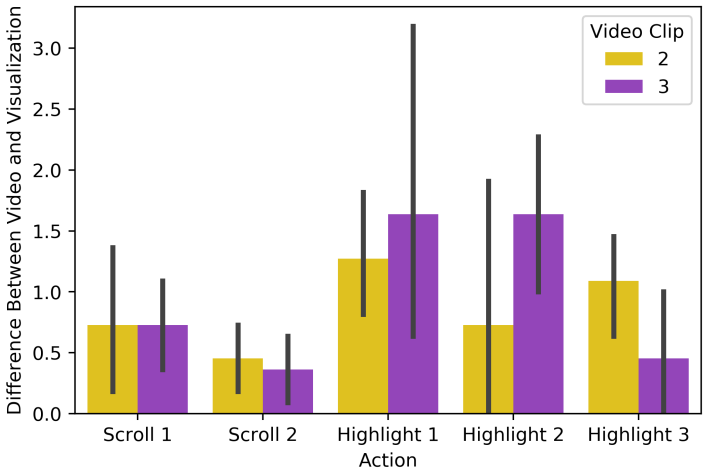
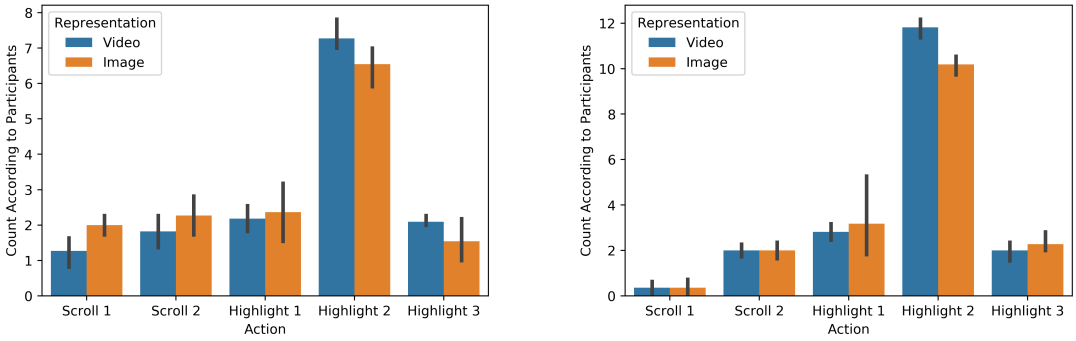


Figure 4.18: The absolute difference between the count for each action when the same evaluator coded the video and visualization of each clip.

There was no action that all 11 evaluators counted the same way in both video and visualization representations of a clip. However, as shown in Figure 4.18, the average absolute difference between the count for the video clip and the count when viewing the visualization was never more than 2. Since all actions had integer counts, this is a small difference. Even in extreme cases, the difference was only once larger than 3, and that was for the action with the highest count in either clip as shown in Figure 4.19.

Figure 4.19 shows the average count for each action in each representation. This information gives context to the results displayed in Figure 4.18 and shows if any actions are repeatedly assigned higher or lower counts when coded from the visualization. The only actions that had the same average count in both representations were the scrolling actions in clip 3. This was not surprising given the ambiguous nature of the actions participants were asked to count. For both video clips, the difference between participant answers when viewing the visualization representation of the clip and viewing the original video clip was usually within a standard deviation. The median responses for the visualization and video representations matched for 3 out of the five actions in both clips, though which three was not the same.



(a) Video Clip 2

(b) Video Clip 3

Figure 4.19: Participant counts of each action based on whether they were viewing visualizations of the clip or the screen recording video.

On average, participants completed the task of counting actions faster when viewing the visualizations than when viewing the screen recording for both clips. Only 2 of the 11 participants took longer to count actions when viewing clip 2 as a visualization than when viewing it as a video. Many participants reported that the visualizations became easier to read with practice. The visualization of clip 2 was the first coding task after the instructions, so it is likely these participants were still getting used to the visualization representation. All participants completed the count for video clip 3 much faster when viewing the visualization. Video clip 3 was 4 minutes and 30 seconds long while video clip 2 was only 3 minutes and 20 seconds. The larger time difference for video clip 3 suggests that the longer the screen recording, the more human coders benefit by working from the visualization. The time spent in each condition is shown in Figure 4.20.

Five of the 11 participants who felt they understood the visualizations mentioned that the highlighting visualization was difficult to read because the pixels were too small or there was not enough contrast between the colors. The third clip had a lot of short highlights and selections, making it likely that many participants struggled to read the highlighting visualization in this case. It is possible that a visualization that re-assigned the highlight colors to a set with more contrast would have gotten better results.

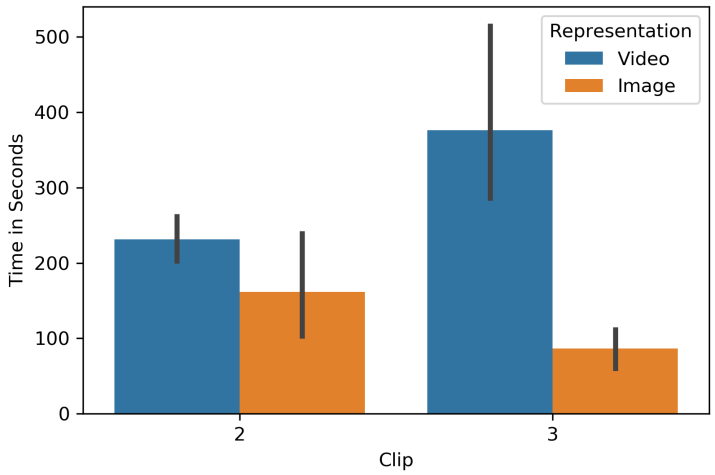


Figure 4.20: The amount of time users spent on each video clip in each condition.

This study shows that human coders can save time by using visualizations instead of watching screen recordings, but it is important that the visualization be easy to read and all coders be trained in how to read them and use them effectively. Typically, human coders discuss what they are coding with each other and iterate on the definitions of what they are coding until sufficient agreement is reached. The participants in this study were not allowed to do this, so it is probable that coders in a real world study would have less variance in their answers.

4.8 Running Time Tests

The time it takes TESS to process a video was calculated by running TESS on 388 videos that range between four seconds and 26 minutes in length and timing each step. The results showed that, the amount of time TESS takes to process a video is based more on the number of time windows it finds than the length of the video. On average, TESS takes 10.5 seconds to process each time window. Most of the videos analyzed in this experiment had less than 14 time windows per minute, but some users had a lot more. The video with the most time windows had more than 2.8 per second. On average, TESS found 19.5 time

windows per minute for videos in this data set, so a researcher who wanted to analyze a 30 minute screen recording could expect TESS to take 1.7 hours. The screen recorder used to capture these videos did not record the mouse, so the number of time windows detected in this test may be lower than would be the case with a screen recording in which the mouse was wiggled a lot.

The most time intensive part of TESS's pipeline is running the OCR engine which on average takes 5.3 seconds per frame. Thus, this accounts for half of TESS's running time. Cleaning up the OCR output is the next most time intensive part. This takes 2.9 seconds per frame and accounts for 28% of TESS's running time. The finding transitions step takes up 14% of TESS's running time, and is the third most time intensive step. Based on the results above and the evidence in Section 4.1 that TESS finds more time windows than necessary, it would be worth taking more time on this step to prune out unnecessary time windows. The other 8% of TESS's running time is predominantly spent preparing frame images for the OCR engine (1.9%), finding each word's color (1.4%), and making visualizations (1.4%). Aggregating actions across time windows and finding the document in the corpus to use in the correction step are the fastest parts of TESS's pipeline.

The version of TESS timed for this test resized each frame image to make them easier for the OCR engine to read. To detect colors in each frame it used the thresholding method of partitioning the text and background colors instead of the clustering method. As discussed in Sections 4.2 and 4.5.2, resizing frame images may not be a necessary step and for some highlight color pairs, thresholding will not find the highlight colors while clustering will. Based on the relative times it took to run thresholding and clustering for the test described in Section 4.5.2, clustering takes is 16 times longer than thresholding. Thus running TESS with clustering instead of thresholding would lengthen the runtime by 21%. Getting rid of the image preparation step would decrease the runtime by 1.9%.

TESS's running time is 3.4 times the length of the video it is analyzing on average. The running time is based on the number of time windows found, so the screen recordings of

more active users take longer to process. Fortunately the action aggregation and visualization steps run quickly, so researchers who use the the interactive editor to fix errors do not have to wait long for each iteration step. TESS is slower than real time, but much faster than humans performing the same tasks as demonstrated in Section 4.1.

4.9 Evaluation Summary

The evaluation tests described in this chapter demonstrate that TESS can track micro text movements and interactions, works with all digital reading environments, and process large data sets faster than human coders. Based on the above experiments, TESS examines twice as many frames in each video as necessary, but does not appear to miss transitions that are consistently noticed by human reviewers. The initial results from OCR engine have several errors, but most of these are fixed during the correction step. The remaining errors all occurred near the edge of the frame or in text that which has difficult colors for the OCR engine, suggesting the interactive editor can do more to highlight areas where errors are most likely to occur. Our experiment on TESS's document matching step shows that TESS can be used in studies with a fairly large stimuli corpora provided the researchers are willing to copy all the text to text files and ensure line breaks are in the proper places. TESS's highlight detection also works well in most cases. The cases where highlight detection failed all involved more than two colors being present in the area that a word was displayed. It may be possible to fix these errors by clustering the pixels into three groups instead of two.

The importance of accuracy will not be the same for all reading experiments. For researchers examining highlighting behaviors, it is important that words are correctly identified when they change color, but if words are incorrectly identified in other places, it may not affect the study. In these cases, human reviewers can double check TESS's results for transitions in which words change colors without reviewing the entire recorded video. Similarly, in a study that uses eye tracking, the words that matter are the ones that the par-

ticipant looked at. Human coders can be asked to double check words that were looked at according to the eye tracking data without needing to review the entire output. In studies of navigation behavior, it is not necessary that all words be identified correctly to detect if the user scrolled or changed the document being viewed.

Chapter 5

Reading Experiments that use TESS

This chapter presents two case studies of using TESS for educational reading research. These studies demonstrate that TESS is useful and present a holistic evaluation. The experiments presented here are not meant to represent all types of studies that TESS might be used for. TESS can monitor a lot of actions, but analyzing those actions and contextualizing them in the existing literature takes time. The studies presented in this chapter both analyze a single data set collected in the spring of 2017. The first study examines scrolling behavior as it relates to learner comprehension and the second explores how participants used zooming.

Both studies use a dataset collected by Amanda Goodwin and her research staff in three middle schools around Nashville, TN. I used TESS to turn the collected videos into CSV files as described in Section 3.5.2. For the scrolling study, these CSV files were analyzed by Sun-Joo Cho who is a faculty member in statistics. The zooming study uses less complex models, and for this study I analyzed the CSV files myself. More detailed information about the dataset collected by Dr. Goodwin is presented in Appendix A. The scrolling study has been published in the International Conference on Learning Studies [33].

The goal in building TESS is to create a tool that helps reading researchers quantify their data. Specifically, this tool should be easy to setup with any stimuli, track text movements and interactions, and scale for larger studies. The studies presented here demonstrate that TESS fulfills these sub-goals and provides value in reading research holistically. These studies show that TESS not only allows researchers to analyze their data in ways not possible without it but also that TESS enables researchers to get more out of each data set they collect and re-analyze previously collected data in new ways. They also illustrate how researchers can work around TESS's limitations by concentrating on line level statistics and

studying reading environments that TESS finds easier to analyze. Other plans for analyzing the data set used in the scrolling and zooming experiments presented here are discussed in Section 5.3.

5.1 Scrolling

In the digital era, it is important to understand how design choices in digital reading environments affect comprehension. This study examines how scrolling, the most popular method of reading long text passages on digital devices, correlates with comprehension. Previous research has found some evidence suggesting that scrolling is worse for comprehension than paging. Research comparing reading on paper to reading on digital devices found that there is no significant difference when the stimuli fits on one screen, but students do worse on digital devices when the stimuli takes up more than one screen [107, 193]. Other research suggests comprehension is better for digital readers using a paging interface than those using a scrolling interface [167, 185]. These findings are in line with research that shows readers use spatial markers to remember what they have read [16, 181, 209]. When these spatial markers move, as happens while scrolling, readers cannot remember what they read as well, or find previously read passages as quickly.

This study is the first to examine how scrolling frequency is linked to comprehension. If scrolling hurts comprehension, then learners who scroll frequently while reading should perform worse on an open book quiz than those who move the text less often. For this study, researchers recorded the screens of 381 5^{th} to 8^{th} graders in the United States while they read a passage on a laptop and then took an open book test. Scrolling was measured by TESS. If the text moved by more than 4 pixels between two frames, that transition was counted as a scroll. This study found that students who had no preference between digital and paper reading performed worse when they scrolled frequently, but scrolling had no effect on the learning of other students.

5.1.1 Background

In the most recent synthesis of comparisons between digital and paper reading, Singer and Alexander [193] conducted a survey of 15 studies occurring between 2001 and 2017 that compared digital and paper reading for texts of different lengths. Seven of these studies used a stimulus which was 500 words or more, and eight used a text which was under 500 words for their stimulus. The eight studies which used shorter texts either noted there was no difference in comprehension between digital and paper reading, or found that digital reading was better for comprehension. In contrast, six of the seven papers which used longer texts found that students comprehended better in the paper condition. The one exception exclusively studied second language learners and noted that the dictionary lookup functionality in the digital condition was popular. Singer and Alexander hypothesized that the reason digital readers consistently did worse than paper readers on comprehension tests for long passages but not short passages is that the most popular method of navigating long text on the computer screen, scrolling, negatively affects reading comprehension. This hypothesis is supported by a 1997 study by Piolat et al. that compared scrolling and paging as methods of interacting with digital text. That study found that the summaries written by undergraduates who paged through a digital document were more coherent than those assigned to navigate the same document with scrolling [167]. Part of these results may be explained by the poor user experience of scrolling in many applications in 1997. A study of 10 computer scientists in 1997 found that scrolling mechanisms were very slow and users reported being annoyed that to scroll they had to click on a small scroll bar and that the display did not refresh fast enough to show them their position while they were scrolling [156].

More recent studies suggest there are some negative effects from scrolling, but they are not as strong as those Piolat et al. found in 1997. In 2007, a study of Swedish participants found that those who had to use a mouse pointer to scroll while reading reported more stress and mental fatigue than those who had to use the keyboard arrows to page

through the document [208]. However, there was no difference in how those who navigated by scrolling and those who navigated by paging performed on comprehension exams after reading. In 2009, a study of American undergraduates found that among students who were either less familiar with the subject matter of the stimuli or less familiar with reading webpages, comprehension test scores were higher if they clicked through digital pages than if they scrolled while reading [185]. For students familiar with the stimuli or familiar with reading webpages, there was no difference between navigating by scrolling and navigating by paging. A 2011 study of Polish adults found that participants who paged through paragraphs and those who scrolled through the document performed equally in a recall test, but those assigned to the scrolling condition took longer to read the document [113].

Together these studies suggest the impact of scrolling on reading processes and comprehension may be decreasing as the user experience improves and familiarity with scrolling increases. The question explored in this study is whether scrolling in today's digital age would impact reading or whether perhaps familiarity with this process has reached a point where it no longer influences comprehension and memory. Additionally, each of the studies above compared the occurrence of scrolling to a different behavior (i.e., page turning or clicking through pages). This assumes all scrolling behaviors are similar and have a similar impact on reading. Because readers vary in how they scroll, this study examines the quantity and directionality of scrolling as it relates to comprehension.

One of the reasons scrolling may affect reading, is that semantic memory of text is linked to spatial memory. This was discovered by studies like Rothkopf's 1971 experiment, which found that substantive memory of sub-passages in a document was directly related to incidental information about the spatial location of the sub-passage (i.e. which page it was on) [181]. More recent studies report similar findings, suggesting the arrival of digital age has not changed readers' reliance on spatial mapping of content. A study of French speakers in 1994 found that after reading a sentence on a computer, subjects could accurately point to where on the screen each word in the sentence had been displayed [16].

In 2007, Wegner and Inhoff used an eye tracker during a reading session [209]. They asked readers to look at words they had just read and found that subjects' eyes immediately found the words on the screen without needing to re-read the passage. These experiments suggest that without the support of spatial mapping, the cognitive load on readers is increased. Since text in a scrolled document doesn't have a fixed position like text on a page, readers who scroll more may perform worse on comprehension tests because they are forced to remember content rather than map content. In contrast, readers who scroll fewer times, like those who only scroll when they must access additional text, may have a lighter cognitive load because they may still be able to map content onto space.

The question remains, though, as to whether readers' increased familiarity and quantity of digital reading experiences have changed how they spatially map content in digital text. For example, a digital reader may have developed compensatory strategies for dealing with the fluid nature of digital texts. Accustomed to scrolling, a reader may map content onto other content, which also moves, like an image or heading providing a constant location in relation to the other content in the text even as scrolling occurs. This may provide a similar decrease in cognitive load as mapping content on space within a physical page. Additionally, it may mitigate the negative impacts of scrolling found in earlier studies of digital reading.

All this assumes, though, increased familiarity with digital reading behaviors like scrolling, which have gained a lot of popularity since the late 90s. A study of 15 German university students in 2010 found that most participants preferred to read a small area of the screen and scroll the text they were reading into that area to reading the whole screen and then scrolling to the text [38]. When questioned about their web reading behavior, experienced users have reported that they use scrolling to move distracting ads off their screen while reading webpages [90]. Perhaps the best proof that user interfaces for digital text are improving and gaining popularity is that more users are opting to read long forms of text on digital devices instead of printing them compared to readers from a decade ago [70].

The goal of this study is to investigate scrolling in today's digital era. This is done by examining the variability of scrolling behaviors and whether scrolling impacts reading comprehension or is an equivalent form of navigation. If scrolling creates more of a cognitive load, then learners who scroll more should perform worse on comprehension and recall tests than learners who scroll the text a whole screen's worth only when they must.

There have been a few studies which counted the number of times a participant scrolled, but none have been conducted on the scale described here and none have tied the total amount of scrolling to performance. For example, in 2001, six American academics were recorded searching through 561 documents to find which contained relevant information for a search query [105]. Researchers recorded how often users scrolled in each document by recording the number of times they clicked on the scroll bar. They found that there was no difference between the number of times users scrolled when viewing relevant documents and the number of times they scrolled while viewing irrelevant documents. In 2016, Freund et al. recorded the screens of 41 Canadian participants and counted the number of times they scrolled to the bottom to check the length of the article they were reading [70]. Readers checked the length of the document they were reading more often when the stimulus was a typical webpage with distracting advertisements and pictures than when it was stripped of extraneous content. This suggests scrolling behaviors differ, but does not link such behaviors to comprehension, which is the goal of the current study.

5.1.2 Method

The screens of middle schoolers were recorded while they read a website. A more complete description of this dataset is presented in Appendix A. TESS used these screen recordings to calculate when students scrolled scrolled. Once scrolling was calculated for all participants, it was linked to comprehension using a statistical model which took other factors such as prior knowledge into account. I ran TESS on the screen recordings and then gave a CSV describing scrolling behavior for each participant to Sun-Joo Cho who was

responsible for building the statistical model.

The participants were 381 fifth to eighth graders (N=87 fifth graders, 78 sixth graders, 83 seventh graders, and 132 eighth graders) who were learning in the classrooms of 11 teachers across 3 schools in Nashville. Eighth graders were over-sampled because the stimulus was a passage designed for 8^{th} graders by NAEP (the National Assessment of Educational Progress). Demographic data was collected for 369 of the participants from the school board. Two hundred and nine (56%) of the students were female. 165 were White, 157 were Black, 34 were Hispanic, 12 were Asian and 1 was an American Indian. According to the 2010 census, this sample had about 12% more Blacks and less Whites than Nashville's distribution, but otherwise matched city-wide demographics. There were 87 students noted as economically disadvantaged and 66 spoke a language other than English at home.

All students did a mix of digital and paper reading as part of their regular classwork according to their teachers, though no classroom used one-to-one digital devices in class. All digital reading was recorded by iMotions [98]. The recording contained an estimate of where the participant was looking aligned with the digital content being viewed, but only the screen recording with processing from TESS was used for this study.

For the content, an NAEP passage [178] was divided into two parts with the dividing point chosen due to the natural end of a section. Therefore, the first part had 434 words and the second part had 674 words. Each participant read one part on paper and one part on a laptop with the order of the reading randomly assigned. Students were interviewed individually in a quiet classroom in their school. Student familiarity with the topic and preferences regarding digital and paper reading tools were measured using a pre-test. After reading both sections, participants took an open book post-test in which most of the questions referred to specific sentences or paragraphs in the stimuli. For a complete description of the data collection process and a list of the questions on the pre- and post- tests, see the Appendix.

When students started the digital reading, the researcher showed them how to use the highlighting, dictionary lookup and touch screen tools which were available in the digital environment. These tools change between devices and platforms, making it unlikely all students were already familiar with the tools, though their mechanisms were the same as similar tools found in PDF viewers.

Scrolling was calculated by TESS using the process described in Section 3.4.1. Due to noise in an image, the OCR engine doesn't always put bounding boxes in exactly the same place, so a transition only counted as a scroll if the bounding boxes moved vertically by at least four pixels. A test of the accuracy of this process is presented in Section 4.6.1. TESS was able to calculate scrolling metrics for all but three of the students. The screen capture program crashed for two of the students, and the OCR engine could not identify the text for one student because they highlighted the entire text dark blue. The scrolling evaluation test described in Section 4.6.1 shows that TESS's scrolling detection for black text on a white background is reliable, so TESS's results were used with confidence. Both of the passages used for this study were too long to fit on a single screen, so all participants had to scroll down to see the full content.

This study used three scrolling statistics for each participant:

- Total number transitions in which the participant scrolled
- Number of *up* scrolls
- Number of *down* scrolls

The latter two statistics were used to check whether scrolling up to possibly review the document had a different relationship with comprehension than scrolling down. One hundred and forty-one students (37%) never scrolled up after the experimenter showed them how to scroll and 241 (64%) had less than five frame transitions in which they scrolled up. For perspective, the medium number of frame transitions in which a student scrolled down after the instructor gave them control of the computer was 14. This meant that for most

participants, the number of transitions in which they scrolled down and the total number in which they scrolled ended up either being the same or very close.

Since the laptop used to display the text was a touch screen, a lot of participants opted to zoom in to see the text or picture better (109 of the 378 students TESS extracted data for zoomed in at least once). Due to the sensitivity in multiple directions of touch screen interface, it was very easy for students to accidentally scroll while zooming. For this reason, if the zoom level before and after a transition was not the same, that transition was not counted as a scroll even if the text moved vertically.

Scrolling was linked to post-test results (i.e., comprehension) using generalized linear mixed models. Each model was fit using the Laplace approximation for maximum likelihood. The model took pre-test scores, grade level, section read digitally, reading modality preferences and ethnicity into account as covariates. I was responsible for running TESS and Sun-Joo Cho used TESS's output to build the model. In addition to directly testing whether the number of times a subject scrolled had a relationship with comprehension, the model considered the effects of holding these variables constant. There were eight students who did not have a complete post-test and therefore had to be left out of the model. With the three students whose video recordings could not be processed, the final model was limited to 370 participants.

5.1.3 Results

Among students who had no preference between reading digitally versus on paper, there was a significant negative relationship between scrolling down and comprehension, controlling for the other covariates in the model. This group represented 39% of the subjects and was larger than either the set of students who said they preferred reading on paper (24%) or the set who said they preferred to read on digital devices (37%). The students without a preference between digital and paper were disproportionately likely to be non-white, but otherwise seemed demographically equivalent to the rest of the class. The analysis did not

find any other relationships between scrolling and comprehension.

There was no relationship between scrolling and pre-test scores, reading aptitude or ethnicity. The frequency with which students scrolled roughly corresponded to how much they liked digital environments. Students who preferred reading on digital devices scrolled more than their peers, followed by students who liked both digital and paper reading.

On average, students who scrolled through the shorter section had 27.5 transitions in which they scrolled and students scrolling through the longer passage had 42.7. It seems that about 11 transitions per student are due to the researcher showing them how to scroll at the beginning of the session. If scrolls in the first minute are excluded, the average drops to 16.9 for the shorter section and 31.6 for the longer section. There was a wide range of scrolling behaviors in both conditions. The standard deviation was 19.6 timeframes for the short passage and 29.1 timeframes for the longer passage, with very long tails to the right in both cases.

The amount of time a student spent reading the digital text was positively correlated to how much they scrolled. This correlation was not so strong as to suggest scrolling is be an approximation of time, nor did dividing the number of times a person scrolled by the time they spent reading produce different results. On average students scrolled once every 14 seconds, but most scrolled more often than that (the median was a scroll every 10.5 seconds).

There was a correlation between the number of times a student scrolled up and the number of times they scrolled down. This relationship is shown in Figure 5.1. This relationship persists even if the cutoff for a 'scroll' is increased to 10 pixels.

5.1.4 Discussion

Preferring text to be available in both paper and digital editions is not unusual among adults. A survey of New Zealand college students found that 49.7% preferred that their textbooks be available in both print and online [202]. However, many of the reasons cited

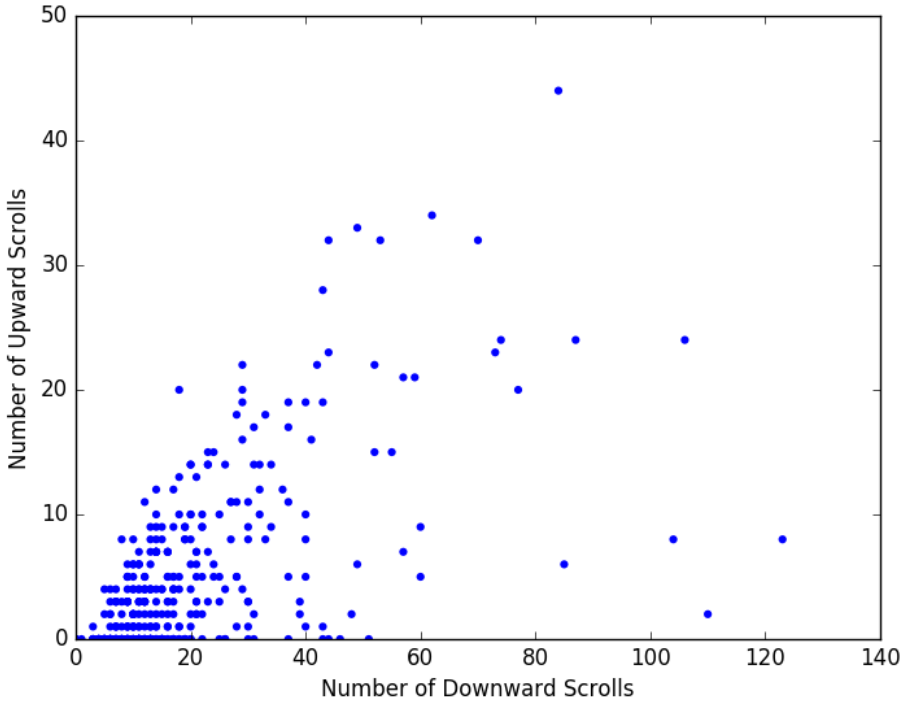


Figure 5.1: Number of upward scrolls plotted against number of downward scrolls.

by adults for preferring materials in two formats such as reading the digital one on the train and the paper one at home and saving money by reading mostly online and printing critical passages do not apply to middle schoolers. This suggests that the two fifths of subjects who did not have a preference between digital and print reading may have not had a chance to develop a preference due to not getting as much exposure to each environment as their peers.

If this conjecture is correct, then it would mean these results line up with Sanchez and Wiley's [185] and support the hypothesis that negative effects from scrolling go away as students get more experience scrolling. It is notable that among this population, only down scrolling had a negative relationship with comprehension, not scrolling up. This supports the spatial hypothesis as an explanation of why scrolling is bad for some students. Frequently scrolling down allows the reader to only read the top line of the screen and thus erase spatial ques such as position on the screen which might have helped their memory.

Scrolling up indicates that a learner took the opportunity to review or checked the length of the text, neither of which would impact their spatial memory.

One way of adding spatial cues while reading from a screen is to move text by a page at a time so that the spatial cues of the screen are re-introduced. Another method would be to train readers to use other spatial markers like text headings and pictures to orient the text they are currently reading in a space which is not dependent on the current placement of the screen. It is notable that learners who prefer to read from paper scroll less than their peers, suggesting they are employing the first strategy and learners who prefer to read digitally scroll more than their peers, suggesting they are using the latter strategy. Learners without a preference may still be using the screen position to remember what they have read but not adapting their scrolling behavior accordingly. Thus, in this group, those who scroll more are not comprehending the text as well. Training these learners to either scroll less often or pick up on spatial cues like headers and pictures might improve their comprehension.

Historians of printed books have noted that the modern method of reading printed books is only a few centuries old. Sixteenth century English bibles were littered with indices and concordance lists which encouraged discontinuous reading much like the links on Wikipedia sites today [90]. Annotations and diaries from the era confirm that many readers did take a discontinuous approach to reading their Bibles. In the intervening centuries, long form printed text meant to be read from beginning to end has become popular and methods of reading it efficiently have been developed. As more educational tools move online, it becomes more necessary to train students in practices which will help their digital literacy, which may not be the same as the best practices for reading on paper. This may include introducing them to alternative navigation techniques or different ways of viewing space.

5.2 Zooming

Zooming is an action that enlarges or shrinks objects on a screen. This action has existed as a method of interacting with some computers at least since the 1980s [14], but it was not until the popularization of devices with small displays and interactive touch screens that researchers noticed enough users zooming to make it a popular area of study [84, 85, 218]. Zooming is still less common than actions like scrolling [82, 84], but researchers have found that it is a strong indication of user interest in studies of web search [4, 124, 137].

This study explores zooming behaviors among middle school readers using a touch screen laptop and how these behaviors correlated with student demographics and learning outcomes. Subjects were asked to read a short passage on a touchscreen laptop at or above their grade level and then complete an open book test on what they read. Each subject was shown how they could zoom the text using the touch screen before they started reading.

5.2.1 Background

Mobile devices popularized zooming by forcing users to use zooming in order to view content that was not designed for the device's screen size. Some webpages which were built to be viewed on desktops are difficult to read on mobile devices without zooming in [124]. Even if a webpage is designed with mobile devices in mind, the difference in font size between languages can lead to the font being too small to see in translations of the content, forcing readers of the translated content to zoom in [152].

The increased use of zooming has also standardized how users zoom. In a 2014 study, Avery et al. discuss some of the issues with zooming via 'pinching' such as users accidentally panning the page while zooming and frequently needing to zoom out and back in to navigate [14]. Other researchers have suggested alternative methods of zooming that do not have these short comings, but in 2014, Avery et al. concluded that 'pinching' had become so ubiquitous that it made more sense to improve how devices computed pinching

actions than to introduce a new method. Other researchers came to similar conclusions after a video of a young girl 'pinching' and 'swiping' the pages of a magazine titled 'A Magazine Is an iPad That Does Not Work' went viral in late 2011 [101].

Even though zooming is becoming more common as touch screens and touch pads become ubiquitous, it is still rare compared to other actions like scrolling [82, 84]. Research on how zooming is used by readers is limited since studies which examine zooming must contain large enough samples to account for the fact that most readers will not zoom.

Experiments that measure zooming during information retrieval tasks have found that zooming is correlated with user interest in the part of the page being zoomed. Agichtein et al. found that they could predict searcher satisfaction with 80% accuracy using scrolling and zooming behaviors [4]. A study by Guo et al. examined replays of users zooming and calculated the correlation between zooming and content relevance. They found that users appeared to be reading the text carefully when they zoomed, but there was no significant relationship between zooming and content relevance [82]. A 2016 study by Lin et al. found that zooming was a strong indicator of interest among readers examining hotel reviews on mobile devices [137].

Zooming can also be used to make text easier to read. A study by Beymer et al. found that readers fixate longer on text that is a smaller font size [26], suggesting smaller fonts are more difficult to read. A 2011 study by Bababekova et al. found that readers using mobile devices tend to hold the devices closer to their faces than they hold text with the same font size when printed on paper [15]. The authors suggest that digital reading environments should default to larger font sizes, since holding back-lit displays close to an eye can damage vision in the long term.

5.2.2 Method

A group of 373 fifth to eighth graders (N=86 fifth graders, 77 sixth graders, 83 seventh graders, and 127 eighth graders) read an article about the passage of the 19^{th} amendment.

The article was a passage designed by the NAEP for an 8^{th} grade reading level [178]. Eighth graders were over-sampled for this reason. Each student read half the article on paper and half on a touchscreen laptop. During the digital reading, the screen of each student was recorded as they read and TESS used these screen recordings to measure zooming behavior. Each students was interviewed individually in a quiet classroom in their school. Student familiarity with the topic and preferences regarding digital and paper reading tools were measured using a pre-test. After reading both sections, participants took an open book post-test in which most of the questions referred to specific sentences or paragraphs in the stimuli.

When students started the digital reading, the researcher showed them how to zoom and scroll on the touchscreen laptop. The researcher also demonstrated highlighting and dictionary tools that were available in the digital reading environment.

TESS was used to calculate zooming behavior. The zoom level for each frame was quantitatively defined as:

$$1 - \frac{height\ of\ the\ current\ frame}{height\ of\ the\ first\ frame}$$

The first frame is the first frame in which the student had control after the researcher demonstrated the tools. Students who did not zoom had a zoom level near throughout the session. Due to noise in TESS's measurements, no student had an entire session in which zooming never rose above 0. The margins in the digital reading environment were large enough that students could zoom in to 40% and still fit the entire width of each line on the screen.

For each student, a 'maximum zoom level' was calculated by finding the largest zoom level that the student stayed at or above for at least a second cumulatively. Some students had very brief zoom 'spikes' which were caused either by the student playing with the screen or measurement errors by TESS. To prevent spikes that were too short for a meaningful interaction from influencing the results, zooms that lasted less than 5 seconds were

filtered out. The larges 'maximum zoom level' observed in any student's session was 52%.

Students were counted as having 'zoomed in' while reading if their maximum zoom level was at least 10%. This study examined demographic and learning outcome differences between the students who zoomed and those that did not. Per line zooming behavior was also computed, and students who zoomed into the lines that answered a question were compared to students who zoomed while reading the same section of the article but did not zoom into the relevant lines.

5.2.3 Results

Overall, 89.5% of students did not zoom in. Zooming was more common among students reading the second half of the article which had 1.5 times as many lines as the first half. Of the students who read the first half of the article digitally, only 8.6% zoomed in at least 10% while 14.5% zoomed in while reading the second half. There were no significant demographic differences between zoomers and non-zoomers.

On the post test, zoomers did worse than non-zoomers. They were significantly more likely to get questions wrong at the end of the post-test, particularly if they read the section that answered those questions on paper. This second section was longer and contained just over half the answers for the questions on the post test. Students who read the second section digitally also did worse on the post test if they zoomed than if they did not, but to a lesser extent than students who read the second section on paper and thus could only zoom by holding the paper closer to their face. The difference in performance based on zooming is illustrated in Figure 5.2.

An analysis of which lines students zoomed into found that zooming was more popular for lines in the middle and end of the documents than lines at the beginning. Students reading the first section tended to stay zoomed in leading to more zooms at the end of the document. Some students reading the second section also maintained a zoom level after a few lines, but using zooming to enlarge the image in the middle of the document was

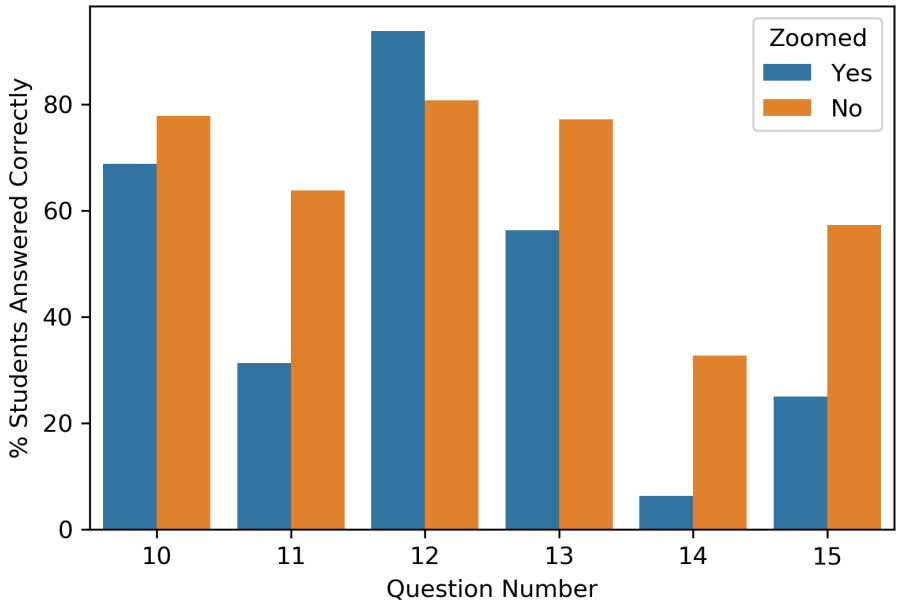


Figure 5.2: The percentage of students who got questions related to the second section correct based on whether they zoomed in at least 10% while reading the first section. The differences are statistically significant for questions 11, 13, 14, and 15.

also popular among students reading the second section. This behavior lead to spikes in zooming around the middle of the document, particularly for zoom levels above 10%. The amount that students zoomed in is shown in Figure 5.3.

Most of the questions in the post test referred to particular lines in the article. A comparison of student performance on those questions based on whether they zoomed into the relevant lines in the article found that students whose zooms included the relevant lines did significantly better on some questions, but did not always outperform students who zoomed into other parts of the article. When the threshold for a zoom to count was raised from 10% to 20, 30 or 40%, the relationship between zoom into a line and answering the related questions on the post test correctly increased, but continued to be insignificant for most questions.

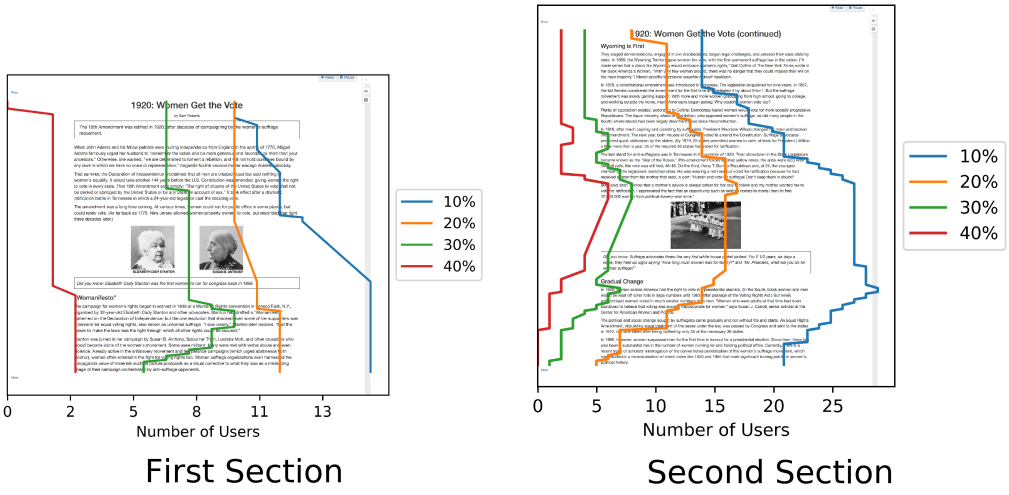


Figure 5.3: Number of students who zoomed in various amounts while reading the first and second section of the article.

5.2.4 Discussion

The results suggest that most students in this study who zoomed in did so to enlarge the text and make it easier to read. Most students who zoomed in do not appear to have paid special attention to the lines they were zooming into since there was no consistent relationship between zooming into a line and answering the related question correctly. The fact that students who zoomed while reading the first section did worse on the questions about the second section suggests that these students find it difficult to read small text. Zooming allows these students to enlarge the text to a size that is comfortable for them.

There was a spike in zooming around the image in the second section but no spike for the images in the first section. The images in the first section display the heads of two suffragettes while the image in the second section shows a group of women carrying a banner. The text on the banner was about half the height of the text in the rest of the article. Thus, students who wanted to read the banner may have done so by zooming in. This suggests that even zooming around images may be explained by text size, which would be in line with the observations on zooming around text.

Some authors have suggested that users may compensate for small text sizes by moving their face closer to the screen rather than zooming [15]. Whether students are moving their faces closer to a screen can be measured with an accurate eye tracker using the distance between a participant's pupils to compute their distance from the screen. Eye tracking data was gathered for this study, but data contained too much noise to use. In future work, eye tracking data can be paired with TESS's output to measure how participants mix zooming and leaning while reading.

This experiment found evidence that among middle school student readers, zooming indicates that the reader may need glasses rather than an interest in an area of the text, which was found to be a motivation for zooming among readers performing information retrieval tasks [4, 82, 137]. This suggests zooming motivation is task dependent. Further research is needed to discover whether the age of readers plays a role in why they choose to zoom.

5.3 Experiments Summary

The experiments described above show that TESS can be used in real world reading experiments to test hypotheses and analyze reading experiment data in new ways. These experiments concentrate on parts of TESS's output that are particularly reliable, but the researchers who collected this data set are interested in using TESS to analyze more of the data set. This data set also contains eye tracking, emotional response, mouse movement and key press log data. As discussed in Section 2.1.5, there is ample evidence that these data sources contain interesting information and are worth analyzing, but working with multiple data sources takes longer since the amount of data is larger.

TESS's document matching was also found to be a reliable output, as explained in Section 4.3. The data set analyzed in this chapter contains several videos of participants reading from one of two articles and then answering a series of questions which are displayed in a paging environment. The experiments on scrolling and zooming concentrate on the reading

portion of each video, and as a result make very little use of TESS's document detection capabilities. In future work, the post test section of each video can be analyzed by using document matching to measure how long each participant spent on each question's page. The post test was open book, so the amount of time participants spent on each question is indicative of how quickly they can recall the location of a passage or the information they read.

The accuracy of TESS's highlight detection varies according to the color of the highlights as demonstrated in Section 4.5. Participants in this data set were able to select text and save highlights while they read. The text selections were white text on a dark blue background and the highlights were black on yellow. The regular text color was black on white. Based on the results of Section 4.5.3, TESS's output for the regular text color and highlight color are reliable, but selected words are mislabeled in about 4% of cases. Fortunately, most text selections include more than one word, so even if some of the text is mislabeled, TESS can estimate when the phrase that eventually became a highlight was first selected.

As mentioned in Section 2.1.4, it has been hypothesized that students learn more from notes if they take longer to make them. This hypothesis can be tested by analyzing what was highlighted and how long participants took to turn selected text into a highlight. Using TESS, researchers can calculate the time participants took to select text and save it as a highlight in the data set used by the studies in this chapter. They can then compare those measurements to participants' performance on test questions related to the highlighted section. Early analysis of this data set has found that there is a positive relationship between making highlights in the digital reading environment and post test scores. It was easier for participants to make highlights in the paper environment, and there is no relationship between highlights made on paper and post test scores.

TESS enables reading researchers to analyze collected data in ways not previously possible. The experiments in this chapter demonstrate that TESS's analysis is useful for

understanding how readers approach educational texts and the effects of various reading approaches. This information can be used to design better digital reading environments which will help readers get the most out of the books of the future. These results can also be used to find the behaviors most associated with positive and negative reading outcomes, which designers can use to build integrated tracking tools that guide learners towards better reading strategies. The experiments in this chapter work with a dataset of middle school readers, but there is no reason TESS could not be used to study datasets of other demographics. In future work it would be interesting to compare the studies presented here to similar studies on other age groups.

Chapter 6

Discussion

TESS is a tool for tracking participants in digital reading experiments. Chapters 4 and 5 describe modular and holistic evaluations of TESS's accuracy and usefulness. This chapter reviews the results of these evaluations and their implications for reading researchers who wish to use TESS. The goal in building TESS is to create a tool that:

- Is easy to setup with any stimuli
- Tracks text movements and interactions
- Scales for studies with more than 20 participants

Existing tools do not fulfill these goals, as explained in Section 2.2. The integrated tools that are popular in knowledge retrieval studies will not work with all stimuli software. Other video analysis tools don't track text positions, and human coders can only be used in small scale studies. TESS accomplishes all three goals by breaking screen recordings of reading experiments into still images, analyzing the text positions and colors in those still images and aggregating this analysis into 'micro' actions that are performed in less than a second. These 'micro' actions include scrolling and changing the color of a word. They can be aggregated into 'macro' actions via computational rules or by having human coders label visualizations in the case of more subjective 'macro' actions.

This chapter discusses the evidence of TESS's accuracy and usefulness presented in Chapters 4 and 5. The focus of this discussion is on identifying the circumstances when TESS is most and least reliable, and what this says about the kinds of studies TESS can be used for. Overall the findings show that TESS's results are accurate and precise when analyzing black text on a white background, and that its output is less accurate, but still potentially useful when less distinct colors are used.

6.1 Accuracy

Chapter 4 describes a set of tests that measure how trustworthy TESS's outputs are. The results of these tests show that TESS works better under some conditions than others, and that it is more accurate for some statistics than others. This section discusses the results of these tests and their implications for researchers using TESS.

The video segmentation test described in Section 4.1 shows that TESS segments the video about twice as often as it needs to. This is better than segmenting the video less than necessary, since that would result in actions going missing. However, making extra segments does slow the program down, so an optimal solution would eliminate these extra cuts.

The difference in the time it takes human coders and TESS to find transitions in a video clip shows that computational tools are much faster for coding micro actions than human coders. Any study that wishes to examine the micro actions of more than one or two readers would benefit tremendously from using computational tools for at least part of the process. While existing computational tools, including TESS are not perfect, using human coders for tasks like video segmentation would be prohibitively time intensive for many reading studies.

The image preparation test in Section 4.2 found that preparing images so they are easier for the OCR engine to work with lowers the error rate before correction, but this improvement does not always result in better outcomes after correction. This experiment can be seen as a proxy for the value of improving the OCR engine used. The word error rate before correction was brought down more than 24% on each frame, but that did not result in an improvement after the correction step. This shows that TESS can achieve good results even when the OCR results are subpar and the best way to improve TESS may not be using a better OCR engine.

TESS's document matching step works well even when the corpus of possible documents is large and contains similar documents, as shown in the evaluation test described in

Section 4.3. The experiments discussed in Chapter 5 all required knowing the location of words at least to the line level. However, as described in Sections 2.1.2 and 2.1.6, there is interest within the research community in how users jump between documents. The fact that TESS's document level detection works well, suggests that TESS would be useful for many studies in this domain.

The evaluation test in Section 4.4 examines both TESS's line matching and word matching steps. The results indicate that the most common error during these steps is lines being removed from the top and bottom of the frame because they did not pass the 60% edit distance threshold to be paired with a line in the correction file. This error is unlikely to affect most reading studies, since readers are unlikely to spend much time looking at the lines at the top and bottom of the frame, particularly when they are only partially visible.

The evaluation tests in Section 4.5 show that more problems arise when the text is colored. Words highlighted certain colors are more likely to be mislabeled by TESS. Highlighted words are important for a lot of experiments, both because highlighting indicates reader interest [17, 213] and because users are likely to concentrate on highlighted words. When possible, researchers can use highlighting colors like black text on a yellow background, which TESS was shown to work well for in Sections 4.5.1 and 4.5.3. In future work, it may be possible to improve TESS's performance on difficult colors by reasoning about word labels across frames, finding ways to recolor the image and thus improve the OCR results, or improving the line and word correction algorithms.

While the evaluation tests in Section 4.5 reveal weaknesses in TESS's ability to correctly assign text labels to words that are highlighted, they also show that TESS's ability to recognize the colors of words is fairly robust. Color detection failed when word blocks detected by the OCR engine contained more than two colors. Even when words were detected correctly, the color categorization step mixed up some color pairs, but most colors were recognized correctly. Many of the color pairs that TESS struggled with the most are also difficult or unpleasant for humans to read [28, 75, 83, 175]. Easy to read colors, like

black text on a white background, white text on a dark blue background and black text on a yellow background were recognized correctly in over 99.5% of cases, as shown in Section 4.5.3. Most reading environments use these easy to read highlighting colors which TESS quickly and accurately be categorizes using the thresholding method for dividing colors. In cases where a larger set of highlight colors is necessary, the clustering method is slower but more accurate.

A version of TESS that can detect more than two colors in a word block would be useful both for cases where the OCR engine fails to separate words properly and for studying how often readers select parts of words. When gathering the middle school reading data set used for the experiments in Section 5, researchers observed many students selecting the same segment multiple times before turning it into a highlight because the first selection did not start at the beginning of a word. If TESS could detect partial word highlights, it could be used to detect how often readers edit their highlights after making partial highlights.

The action coding evaluation tests in Section 4.6 show that the error rate for aggregated data does not always correspond to the error rate at the frame level. The screen position is calculated by considering all detected line positions within each screen. The result is that the aggregated scrolling data is very accurate, outside of a few small wiggles. As expected, the frequency of scrolling events is inversely proportional to the length in pixels of each scroll.

The comparison of aggregated highlighting reports in Section 4.6.2 found that frame level errors are more likely to affect highlighting statistics than scrolling statistics. Since highlighting causes more labeling than color detection errors, general statistics such as 'the total number of words that were highlighted' do not have as many errors as statistics specific to a given word, such as 'was the first word of the second paragraph highlighted'. In the test described in Section 4.6.2, 'the total number of words that were highlighted' is off by 2%, while 4% of the words marked highlighted are errors, 2% of the words that were highlighted went undetected, and 17% of the words that were correctly marked highlighted

were not labeled highlighted for the correct times.

In some cases, researchers are interested in discovering whether readers highlight specific words in the document and how long they left those words selected. At the moment, TESS's aggregated highlighting information has too many errors to precisely and accurately answer these questions. However, TESS is useful for studies that require less precision. The highlight labels are usually accurate to within a line, so TESS can track whether readers highlight short phrases more accurately than words. Studies that don't focus on specific parts of the text but instead look at how often readers make and delete highlights are more likely to find that TESS is sufficiently accurate.

Section 4.7 tests the claim that TESS's outputs can be used by human coders to code more subjective 'macro' behaviors faster than watching the screen recordings. The coders did not uniformly agree about the counts for any of the 5 actions either when viewing the video representations of the two sessions or when viewing the visualizations, indicating that the actions coding were indeed subjective. The average count for each action in each representation was usually, within a standard deviation of the average count for the other representation. Since this was an evaluation test, coders were not allowed to talk to each other to reach a consensus regarding how actions should be counted. In a real-world study such discussions are the norm, and coders rarely agree on definitions without any discussion [22, 25]. Thus, it is reasonable to assume that in a real world study the agreement between coders would be higher.

Coders were able to code the actions faster when viewing the visualization representation for both examples. Most coded the visualization shown last faster than the first visualization shown, despite the fact that most counted more actions in the last visualization. This supports the hypothesis that coding the visualizations gets easier and faster as coders become more familiar with the visualizations. One of the recruited coders was unable to code the visualizations because they could not understand the instructions. This was the only recruited coder without a STEM background, which provides further evidence that there is

a learning curve to understanding visualizations of TESS's output, and this task is easier for coders with a mathematical background. Based on these observations, researchers using visualizations of TESS's output to code 'macro' actions may want their coders to have input on the design of the visualizations used to minimize confusion.

The analysis in Section 4.8 demonstrates that to process a video of length x TESS's running time takes $3.4x$ on average, though the actual running time is based more on how active the user is than the length of the video. This can be sped up somewhat for researchers who have access to more computing resources since the most time intensive steps in TESS's pipeline (analyzing frames) can be done in parallel. TESS is slower than most integrated tools which usually log information in real time, but much faster than human coders. Since TESS is run after data has been collected in video format, it is not necessary that it run quickly to be useful for reading experiments. Researchers often spend months collecting and analyzing data. Once data is collected, TESS can be run on a lab computer while researchers discuss which 'macro' actions to analyze.

These internal evaluation tests show that TESS works well for dark text on a light background, and can handle common highlight colors like yellow. Its highlight detection makes the most mistakes when the highlight color is similar to the color of the boarders and menus, and its text detection makes errors when the background is dark. The document and scrolling detection results are accurate and robust. For some studies (examples discussed in Sections 2.1.2, 2.1.3 and 2.1.6) these modules are enough to collect meaningful data. Researchers that use all of TESS's output can mitigate errors by using stimuli that TESS is better at processing, or fix errors using the interactive editing tool.

6.2 Usefulness

The strongest evidence that TESS can help researchers is that it has been successfully used to study real world reading behaviors. This evidence is shown in the experiments in Chapter 5. These experiments are not intended to demonstrate the full breadth of studies

that TESS can be used for. As discussed in Section 5.3, the researchers who collected the data set used in Chapter 5 plan to use TESS to investigate highlighting and question answering behaviors. As TESS is used on more data sets, it will hopefully be used in ways its author has not dreamed of.

The scrolling experiment described in Section 5.1 uses TESS to study the affect of scrolling on reading comprehension. The results show that using short-frequent scrolls has a negative affect on comprehension for a subset of students who do not have a preference between digital and paper reading environments, but doesn't affect comprehension for the majority of students in the study. Student familiarity with digital reading environments was not measured, but since preferences are developed after exposure, it seems reasonable to hypothesize that the students who did not have a preference may have been less familiar with digital or paper reading environments, and thus less trained in how to scroll without influencing comprehension. This result has implications for the scrolling hypothesis discussed in Section 2.1.2, and has been published at the international conference on learning sciences [33].

Section 5.2 describes an exploratory study of zooming behavior among middle school students. The study found that while most students don't zoom, those that do have worse test performance when answering questions based on what they read in a paper reading environment. This indicates that many of the students who use zooming may be doing so to compensate for short-sightedness. However, since participants who zoom into the parts of the stimuli that answer questions on the post test do not consistently perform better on those questions, it appears that students who use zooming may not be getting as much out of it as they could. Further research is needed to confirm these hypotheses. If it is confirmed that students are using zooming to compensate for short-sightedness, this would have several practical applications. For instance teachers could teach students how to effectively use zooming and students observed zooming frequently could have their eyesight tested.

One of TESS's goals is to be easy to setup with any stimuli. Since the existing tools

for computationally tracking text in reading environments are incompatible with many software packages, TESS is easier to setup simply because it can be used at all. However, TESS is harder to use than commercial research tools since it does not have a rigorously tested interface. The amount of technical expertise varies between reading research labs, and labs without any programmers on staff will find it difficult to work with TESS. Fortunately, TESS runs on previously collected data, so labs that cannot afford to keep programmers permanently on staff can hire programmers to analyze their data using TESS after data has been collected. As mentioned in Section 2.1, tools are only adopted when the benefits they provide outweighs the cost of learning how to use them. Hopefully a future version of TESS will be easier to learn than the current one. For the moment, TESS uses its large repertoire of functionality to attract new users.

The studies in Chapter 5 demonstrate that TESS is useful for reading experiments. The tests in Chapter 4 found that despite the errors detected in Chapter 4, TESS's output is reliable and useful for studying current questions on how learners read. Hopefully TESS will become more accurate with further development, but it is just as important to continue exploring how reading research can be advanced with imperfect tools. Chapter 5 only contains two studies, but Section 2.1 discusses many more areas where TESS's output can be applied. In all of these areas, it would be foolish to wait for a tool that never makes mistakes.

6.3 Discussion Summary

The experiments in Chapter 4 show that TESS's accuracy varies based on attributes of the reading environment, but that it works well for many popular reading environments. As discussed in Section 2.2, other computational tools for tracking the position of text in a screen recording do not currently exist. TESS is an early prototype of this type of tool, and has several limitations. A discussion of how TESS can be extended in future work is presented in Chapter 7, the current version works best for studies of black text on light

backgrounds. The experiments in Chapter 5 demonstrate that TESS's outputs are useful in practice.

As mentioned in Section 3.7, one of TESS's strengths is its modular design. This allows researchers to use parts the of TESS that are applicable to their studies and ignore parts that aren't. Both of the studies in Chapter 5 were carried out without color detection. As mentioned in Section 5.3, the researchers that collected the data for Chapter 5 want to use color detection in future work, but they were able to run studies on their data before using the color detection step.

Chapter 7

Concluding Remarks and Future Plans

TESS is a tool that can be used with any screen recording, tracks text accurately for the most common colors used in reading environments, and scales for studies with hundreds of participants. It tracks text movements using video recordings from the participant's point of view. Since screen capture technology is widely available and easy to setup, TESS opens fine grained tracking capabilities to many reading researchers who do not have the resources for existing text tracking technology. Experiments comparing TESS to human coders and other tools have demonstrated that it is reliable for high contrast English text. In cases where TESS's text labels are less reliable, statistics such as the number of highlighted words and the amount of scrolling can still be collected.

There is ample evidence that human readers make heavy use of context, as indicated by a number of studies. When reading a sentence that contains a couple letter transposition errors, many readers will fail to notice the typos at all if the transposed letters are not at the beginning or end of the word and occur close to each other in the correctly spelled word [166]. Similarly, studies have shown that readers do not always notice missing letters [187]. This is not to say that permuting words has no effect. A 2006 study of 30 university students found that transposing internal letters in 40% of words in a sentence decreased reading speed by 12%. Transposing letters at the end of words decreased reading speed by 26%, and transposing letters at the beginning of words decreased reading speed by 36% [177]. Readers had high comprehension scores regardless of how many letter transposes were in the text, but they had more eye fixations and each fixation lasted longer while they were reading permuted text. The practice of relying more on context than letter order appears to start early for human readers. In 2015, a study comparing elementary school readers to adults found that both groups could understand words with their internal letters

transposed [165].

Unfortunately, researchers are still unsure how human readers use context when reading, and how computational readers can use and represent context is still an open question. The solution used by OCR researchers has been to use probabilistic linguistic models in post-processing [100]. These models are often difficult to build and don't generalize to other languages and contexts easily. It is often overlooked that in many situations much more precise and easy to use information about the text may be available. TESS demonstrates one method of using information about the stimuli to create stronger computational readers. The text of a reading experiment and the highlight colors used are both easy to input and don't require extensive linguistic knowledge of the language being used. In future work, researchers may find more ways to use contextual knowledge such as the locations of links and the rules for changing highlight colors to improve optical text tracking systems further.

TESS has several limitations. It does not detect multiple highlight colors within the same word block, it requires researchers using it to enter the stimuli text with line breaks that match the screen recordings (it does not work if the line breaks change during the screen recording), and it does not have a graphical interface. Another limitation is TESS's running time which is about 3.4 times the length of the screen recordings it analyzes as discussed in Section 4.8. For researchers with many hours of screen recordings to analyze, this running time is annoyingly slow. Eliminating these limitations would enable more researchers to use TESS.

Detection of multiple highlight colors within a word block could probably be accomplished by looking at character level bounding boxes. Many OCR engines start by identifying characters rather than whole words [1, 194]. The character boundaries are often lost during post-processing when language models are used to formulate words, but this information could be encoded into the hOCR output. If letter boundaries were known, TESS could use its current approach to find the highlight color for each letter rather than each

word. Highlights don't always fully cover a word, but letters are almost never partially highlighted, and many digital reading environments don't allow readers to partially select letters.

Making a version of TESS that does not require the line breaks in the correction files to match those in the screen recording would help researchers with large corpora. Such a tool would have the same document matching step as TESS but would then move straight to the word matching step. The difficult part in implementing this approach would be maintaining TESS's accuracy without the line matching step. It is possible that this could be accomplished with a word matching algorithm that used a better edit distance, but a more complex model of how words are displayed and covered by pop-ups might also be necessary. Such a model would likely slow down the text correction step. Eliminating consistent lines from the screen recordings would also affect how TESS calculates scrolling and zooming. If participants are allowed to re-size the screen and change the locations of line breaks, then the current method of measuring scrolling and zooming will not work. TESS's output would still contain information on word size and position, but scrolling and zooming cannot be modeled as moving a viewing window around a static image if the line breaks in the 'image' move around.

A version of TESS that does not use the study corpora at all might be useful in certain circumstances. As discussed in Section 4.2, the word error rate after prepossessing is under 10% even without correction. This error rate is problematic for researchers who are precisely measuring gaze or highlights, but scrolling detection does not require all detected words to be correct and many currently available eye trackers are only accurate at the paragraph level [58]. Thus, a version of TESS that didn't correct text labels would still be useful in some studies. Assigning global ids to words that are consistent across frames would be more difficult without correcting the text labels, so the action coding step described in Section 3.4 would need to be rethought. Researchers who study knowledge retrieval in software they do not control, such as those working in the library sciences [32, 77, 97],

have difficulty collecting the corpus of documents used in their studies and might find a version of TESS without text correction useful.

The current version of TESS does not have a graphical interface, is non-trivial to setup, requires users to understand how to represent colors as numbers and how to match line breaks to screen recordings. This limits the potential users of TESS to researchers who are not put off by command line interfaces and lengthy installation commands. At the moment, limiting the potential customers of TESS like this helps to ensure that those who do use it will be able to understand that it is not always accurate. However, in future versions of TESS, making a more user friendly interface will help make this tool commercially viable and expand the potential impact. Other tools used by research communities have seen dramatic increases in adoption after releasing graphical user interfaces [31]. Making TESS faster would also make it more appealing adopted by a wider audience.

Several alternative methods of building a tool like TESS have been suggested by colleagues and reviewers. In some cases these have been helpful and became part of the design described in Chapter 3. In other cases these suggestions turned out to be nonviable. Some reviewers have suggested that scrolling could be tracked faster using optical flow. However, optical flow algorithms rely on the assumption that each object in a frame is fairly unique, particularly within the area it is displayed. This assumption does not hold for most images of text, where multiple instances of a letter look the same and may be displayed next to each other. It has also been suggested that the text labeling errors could be reduced by recoloring each frame prior to running OCR. It may be possible to get more accurate labels by re-running OCR after recoloring words detected on the first pass, but without breaking the frame into areas where a single word is shown, it would be difficult to assign new colors. Even after the original OCR results have been collected, recoloring word blocks that contain more than two colors presents difficulties as discussed in Section 4.5.2.

A tool that combines TESS's text tracking with algorithms like optical flow that track the movement of images or 3D objects could enable researchers looking at video games to

conduct studies similar to those presented in Chapter 5. Video games often contain a large number of heterogeneous objects including 2D and 3D avatars, text, buttons and images. As mentioned in Section 2.2.3, previous work has been done on tracking button presses and other interactions common in graphical environments, but a general tool that can track interactions with all of these objects is likely still a long way off.

There are interesting ways TESS could be extended unrelated to its limitations. For large studies of 'macro' actions, a future version of TESS could train models of each action based on detected 'micro' actions once human coders have labeled a few examples. The number of training examples needed to build a model would vary based on the complexity of the 'macro' actions. If there is evidence of general interest in certain 'macro' actions, models of those 'macro' actions could be trained on one dataset and shipped with distributions of TESS.

In future work, TESS's output could be used to generate more learning examples for training an OCR engine. This would be particularly useful for less studied written languages that don't currently have large OCR training data sets. TESS's current design makes it easy to change the OCR engine used, however a more integrated design with a OCR engine that was updated based on TESS's output might produce more accurate results. An OCR engine that produced very accurate results could be used to detect the locations of typed words. Typing is of interest to many of the educational reading researchers whom TESS is aimed at, so this extension would be easy to market to the same community. Researchers who use keystroke logging tools (which can be run at the operating system level independently of text editing software) can use keystroke logs to correct OCR output for each frame much the same way TESS uses text files. Such a system would be more complicated that TESS since there are many keystrokes that don't generate visible text and some visible characters are made with multiple key-presses such as capital letters, Chinese characters and letters with accents.

TESS was designed to improve studies in digital reading research. It does this by en-

abling researchers to track text positions and colors in from any screen recording. Because the input is screen recordings, TESS can be used with any digital reading software. Digital reading environments are introducing new methods of interacting with text in all aspects of student's lives [126], but how to promote educational and enjoyable interactions in digital reading environments is not well understood [104, 143, 192]. TESS makes it possible to study digital reading environments quickly and reanalyze previously collected screen recordings. The results of digital reading experiments can potentially have a larger impact than studies of analog reading technologies. Readers using digital devices can download updates remotely [13, 111, 173], new digital artifacts are often cheaper to distribute than their paper counterparts [164, 172, 179], and many digital devices are more portable than analog alternatives [211]. The studies in Chapter 5 demonstrate that TESS's outputs are useful in reading studies. As time goes on, more researchers will develop novel ways to use this data to help readers and educators.

Appendix A

Middle School Reading Dataset

This appendix describes the dataset used for the zooming and scrolling studies described in Chapter 5 in more detail. This dataset was collected by Dr. Amanda Goodwin with help from staff in the education department. The statistical model used in the scrolling study discussed in Section 5.1 was built by Dr. Sun-Joo Cho and used TESS's scrolling output. Both the scrolling and zooming studies used data collected from 381 fifth to eighth graders (N=87 fifth graders, 78 sixth graders, 83 seventh graders, and 132 eighth graders) who were learning in the classrooms of 11 teachers across 3 schools in Nashville. Eighth graders were over-sampled because the stimulus was a passage designed for 8^{th} graders by NAEP. Demographic data was collected for 360 of the participants from the school board. Two hundred and one (56%) of the students were female. 161 were White, 152 were Black, 34 were Hispanic, 12 were Asian and 1 was an American Indian. According to the 2010 census, this sample had about 12% more Blacks and less Whites than Nashville's overall demographics, but otherwise matched city-wide percentages. There were 87 students noted as economically disadvantaged and 66 spoke a language other than English at home. These languages included Spanish, Tamil, Amharic, Arabic, Kurdish, Vietnamese, Urdu, Somali, Persian, Rundi, Lao, Twi, Tigrinya, Chinese, French, Tagalog, Mandingo, and Portuguese. Teachers reported that in class students did a mix of digital and paper reading but one-to-one digital devices were not available in the classrooms.

The stimuli was a 2011 NAEP article on women's suffrage (see https://www.nationsreportcard.gov/reading_2011/testyourself_g8_passage_ann.aspx). Participants were told they would be reading an interesting article about women's suffrage. They were asked to read as they would for a homework assignment because they would be asked questions afterwards. Each participant first took a pre-test to asses their existing knowledge of the article's content.

The pre-test also contained questions on how the participant preferred to study. A list of all questions on the pretest is available in Table A.1. Participants could elect to either have the researcher ask the pre-test questions out loud and enter their answers in a Google Form, or fill out the form themselves. Students were encouraged to skip questions on the pre-test they did not know the answer to.

After completing the pretest, participants read half the article on paper and half on touchscreen laptop. Which half a participant read in each modality was determined through random assignment, with participants in condition A reading the first half on paper and participants in condition B reading the first half on the laptop. After reading the article, participants took an open book post-test which was designed to be similar to the reading portion on standardized tests. During the post-test, students were given access to the two halves of the article in the modality they were read in. Most questions in the post-test referred to a specific line in the article, so performance on specific questions could be linked to modality. A complete list of the questions on the post-test and which half of the article contained the answer is given in Table A.2.

The article was divided at a natural break in the text. The first half contained 434 words while the second half contained 674 words. The average sentence in the first half contained 19.48 words versus 14.98 words in the second half. Originally, there was one image in the first half. An image was added to the second half of the article to balance the visual stimuli. The image in the first half showed portraits of two suffragettes, while the image in the second half showed a group of marchers carrying a banner. The text on the banner was small, but legible, which may explain why this image was the most zoomed into part of the stimuli. Both halves required participants to scroll when reading on the laptop, and both were printed on two sides of a single piece of paper in the analog modality.

While reading from paper, participants had access to pens, highlighters and sticky notes. In the digital environment they had access to plugins for making highlights and saving typed notes with their highlights, a dictionary lookup tool and a text readout tool. The

#	Question
1	What do you know about the history of womens voting in the US?
2	What was the constitutional amendment that gave women the right to vote?
3	About how many years did it take after the Declaration of Independence for women to earn the right to vote in every state?
4	When Elizabeth Cady Stanton wrote the Womanifesto, why did she demand equal voting rights rather than other rights like property ownership?
5	Women could run for office before they could vote in every state.
6	Which was the first state to pass a law allowing women to vote?
7	After marches and protests, President Woodrow Wilson supported the 19th amendment.
8	What state played a key role in passing the 19th amendment because it was the final state needed to approve the amendment?
9	Once the 19th amendment passed, the womens suffrage movement took off, with the Equal Rights Amendment being passed shortly thereafter.
10	When was the first election where more women than men voted?
11	If you could choose, how would you choose to read articles?
12	What tools do you use to support your reading on paper?
13	What tools do you use to support your digital reading?

Table A.1: Pre- Test and Survey Questions

readout tool sometimes encountered bugs, but most students did not make use of it at all. The highlight tool failed to save highlights in a few cases due to the save button not getting pushed, but generally worked. The dictionary tool contained links to longer definitions (the basic definition was shown in a pop-up). Students who clicked on these links found themselves in a new tab, but were asked to close the tab and return to the article. This only happened with a few students. The digital tools were modeled for each participant prior to starting the digital portion of the article.

The statistical model used in the scrolling study was a between-subject generalized linear model that looked at the relationship between the each of the three scrolling measures discussed in Section 5.1 (total number of scrolls, number of up scrolls and number of down scrolls) and post-test performance. Pre-test performance, indicated modality preferences and demographic data were used as controls in the model. Modality preference was quantified for the model as paper = -1, digital = 1, and both = 0. The model included grade,

#	Question	Answer Location
1	What do you know about the history of womens voting in the US?	NA
2	What is the main purpose of the article?	NA
3	What was the constitutional amendment that gave women the right to vote in all elections?	Part 1
4	According to the article, women in New Jersey in the 1700s could	Part 1
5	According to the article, what was most surprising about the ”Womanifesto”?	Part 1
6	When Elizabeth Cady Stanton wrote the Womanifesto, why did she demand equal voting rights rather than other rights like property ownership?	Part 1
7	The article says that women in the suffrage movement ”pressed their case state by state.” This means that the women	Part 1
8	Women could vote in certain state elections before they could vote nationally.	Part 2
9	Which was the first state to pass a law allowing women to vote?	Part 2
10	In her book, America’s Women, what did Gail Collins suggest was the reason that the Wyoming Territory passed the first permanent suffrage law?	Part 2
11	After marches and protests, President Woodrow Wilson supported the 19th amendment.	Part 2
12	What state played a key role in passing the 19th amendment because it was the final state needed to approve the amendment?	Part 2
13	Once the 19th amendment passed, the womens suffrage movement took off, with the Equal Rights Amendment being passed shortly thereafter.	Part 2
14	When was the first election where more women than men voted?	Part 2
15	About how many years did it take after the Declaration of Independence for women to earn the right to vote in every state?	Part 2
16	Were there any additional tools that you used either when reading on paper or digitally that you found helpful?	Survey

Table A.2: Post-Test Questions

gender, race, home language, English language learner status, standardized test percentile, and economic status. Most of these variables were coded using a binary scheme. Race was coded as white or non-white, home language was coded as English or different, English language learner status was coded as enrolled in classes or not, and economic status was coded as economically disadvantaged or not. Grade and standardized test percentile were already numeric and were used as is.

BIBLIOGRAPHY

[1] Gheith A. Abandah and Fuad T. Jamour. 2010. Recognizing handwritten Arabic script through efficient skeleton-based grapheme segmentation algorithm. In *Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications, ISDA'10*. 977–982. https://doi.org/10.1109/ISDA.2010.5687062

[2] John Adcock, Matthew Cooper, Laurent Denoue, and Lawrence A Rowe. 2010. TalkMiner: A Lecture Webcast Search Engine. *MM '10 Proceedings of the 18th ACM international conference on Multimedia* 21 (2010), 241–250. https://doi.org/10.1145/1873951.1873986

[3] Mikhail Ageev, Dmitry Lagun, and Eugene Agichtein. 2013. Improving search result summaries by using searcher behavior data. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '13*. 13. https://doi.org/10.1145/2484028.2484093

[4] Eugene Agichtein, Qi Guo, Shuai Yuan, and Eugene Agichtein. 2011. Detecting success in mobile search from interaction Detecting Success in Mobile Search from Interaction. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR, Beijing, China. https://doi.org/10.1145/2009916.2010133

[5] Jason Alexander, Andy Cockburn, and Richard Lobb. 2008. AppMonitor: A tool for recording user actions in unmodified Windows applications. *Behavior Research Methods* 40, 2 (2008), 413–421. https://doi.org/10.3758/BRM.40.2.413

[6] Logan E. Annisette and Kathryn D. Lafreniere. 2017. Social media, texting, and personality: A test of the shallowing hypothesis. *Personality and Individual Differences* 115 (2017), 154–158. https://doi.org/10.1016/j.paid.2016.02.043

[7] Pansy Arafa, Daniel Solomon, Samaneh Navabpour, and Sebastian Fischmeister. 2017. Debugging behaviour of embedded-software developers: An exploratory study. In *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 89–93. https://doi.org/10.1109/VLHCC.2017.8103454 arXiv:1704.03397

[8] Ioannis Arapakis, Mounia Lalmas, and B Barla Cambazoglu. 2014. User Engagement in Online News: Under the Scope of Sentiment, Interest, Affect, and Gaze. *Journal of the Association for Information Science and Technology* 65, 10 (2014), 1–34.

[9] Ioannis Arapakis and Luis A. Leiva. 2016. Predicting User Engagement with Direct Displays Using Mouse Cursor Information. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval - SIGIR '16*. 599–608. https://doi.org/10.1145/2911451.2911505

[10] Ioannis Arapakis, George Valkanas, Mounia Lalmas, and George Valkanas. 2014. Understanding Within-Content Engagement through Pattern Analysis of Mouse Gestures. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* December 2016 (2014), 1439–1448. https://doi.org/10.1145/2661829.2661909

[11] Richard Atterer and Philip Lorenzi. 2008. A heatmap-based visualization for navigation within large web pages. In *Proceedings of the 5th Nordic conference on Human-computer interaction building bridges - NordiCHI '08*. ACM Press, New York, New York, USA, 407. https://doi.org/10.1145/1463160.1463206

[12] Richard Atterer, Monika Wnuk, and Albrecht Schmidt. 2006. Knowing the user's every move. In *Proceedings of the 15th international conference on World Wide Web - WWW '06*. 203. https://doi.org/10.1145/1135777.1135811

[13] Author's Insider Club Admin. 2013. How To Change or Edit A Kindle Book You Have Already Published. (oct 2013). http://www.ebooktemplates101.com/change-edit-kindle-book-already-published/

[14] Jeff Avery, Mark Choi, Daniel Vogel, and Edward Lank. 2014. Pinch-to-zoom-plus. *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14* (2014), 595–604. https://doi.org/10.1145/2642918.2647352

[15] Yuliya Bababekova, Mark Rosenfield, Jennifer E. Hue, and Rae R. Huang. 2011. Font size and viewing distance of handheld smart phones. *Optometry and Vision Science* 88, 7 (2011), 795–797. https://doi.org/10.1097/OPX.0b013e3182198792

[16] Thierry Baccino. 1994. Spatial Coding and Discourse Models During Text Reading. *Language and Cognitive Process* 9, 2 (1994), 143–155. https://doi.org/10.1080/01690969408402114

[17] Vimala Balakrishnan and Xinyue Zhang. 2014. Implicit user behaviours to improve post-retrieval document relevancy. *Computers in Human Behavior* 33 (2014), 104–112. https://doi.org/10.1016/j.chb.2014.01.001

[18] Nikola Banovic, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2012. Waken: Reverse Engineering Usage Information and Interface Structure from Software Videos. In *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. ACM Press, New York, New York, USA, 83. https://doi.org/10.1145/2380116.2380129

[19] Lingfeng Bao, Jing Li, Zhenchang Xing, Xinyu Wang, Xin Xia, and Bo Zhou. 2017. Extracting and analyzing time-series HCI data from screen-captured task videos. *Empirical Software Engineering* 22, 1 (feb 2017), 134–174. https://doi.org/10.1007/s10664-015-9417-1

[20] Lingfeng Bao, Jing Li, Zhenchang Xing, Xinyu Wang, and Bo Zhou. 2015. Reverse engineering time-series interaction data from screen-captured videos. In *2015 IEEE*

22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER). IEEE, 399–408. https://doi.org/10.1109/SANER.2015.7081850

[21] Lingfeng Bao, Zhenchang Xing, Xinyu Wang, and Bo Zhou. 2016. Tracking and analyzing cross-cutting activities in developers' daily work. *Proceedings - 2015 30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015* (2016), 277–282. https://doi.org/10.1109/ASE.2015.43

[22] Tehmina N. Basit. 2003. Manual or electronic? The role of coding in qualitative data analysis. *Educational Research* 45, 2 (2003), 143–154. https://doi.org/10.1080/0013188032000133548

[23] Gal Ben-Yehudah and Yoram Eshet-Alkalai. 2011. The Influence of Text Annotation Tools on Print and Digital Reading Comprehension. In *Proceedings of the 9th Chais Conference for Innovation in Learning Technologies*. 28–35.

[24] Michael Bendersky, Xuanhui Wang, Donald Metzler, and Marc Najork. 2017. Learning from User Interactions in Personal Search via Attribute Parameterization. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 791–799.

[25] Selinda Adelle Berg, Kristin Hoffmann, Diane Dawson, Selinda Berg, ; Adelle, and Kristin ; Hoffmann. 2011. Not on the Same Page: Undergraduates' Information Retrieval in Electronic and Print Books. *Journal of Academic Librarianship* 36, 6 (2011), 518–525. https://scholar.uwindsor.ca/leddylibrarypubhttps://scholar.uwindsor.ca/leddylibrarypub/12

[26] David Beymer, Daniel M. Russell, and Peter Z. Orton. 2008. An Eye Tracking Study of How Font Size and Type Influence Online Reading. In *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction*. BCS Learning & Development Ltd., 15–18. http://link.springer.com/10.1007/978-3-540-74800-7

[27] Nilavra Bhattacharya and Jacek Gwizdka. 2018. Relating eye-tracking measures with changes in knowledge on search tasks. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications - ETRA '18*. ACM Press, New York, New York, USA, 1–5. https://doi.org/10.1145/3204493.3204579 arXiv:1805.02399

[28] Debojyoti Bhattacharyya, Bodhisattwa Chowdhury, Tirthankar Chatterjee, Madhusudan Pal, and Dhurjati Majumdar. 2014. Selection of character/background colour combinations for onscreen searching tasks: An eye movement, subjective and performance approach. *Displays* 35, 3 (2014), 101–109. https://doi.org/10.1016/j.displa.2014.03.002

[29] Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. https://doi.org/10.1103/PhysRevB.62.4273

[30] Mark Bohay, Daniel P Blakely, Andrea K Tamplin, and Gabriel A Radvansky. 2011. Note Taking, Review, Memory, and Comprehension. *The American Journal of Psychology* 124, 1 (2011), 63–73. https://doi.org/10.5406/amerjpsyc.124.1.0063

[31] Remco R. Bouckaert, Eibe Frank, Mark A. Hall, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2010. WEKA Experiences with a Java Open-Source Project. *Journal of Machine Learning Research* 11 (2010), 2533–2541.

[32] Melissa Bowles-Terry, Merinda Kaye Hensley, and Lisa Janicke Hinchliffe. 2010. Best practices for online video tutorials in academic libraries: A study of student preferences and understanding. *Communications in Information Literacy* 4, 1 (2010), 17–28. https://doi.org/10.1017/CBO9781107415324.004 arXiv:arXiv:1011.1669v3

[33] Katherine Brady, Sun Joo Cho, Gayathri Narasimham, Douglas Fisher, and Amanda Goodwin. 2018. Is Scrolling Disrupting While Reading ?. In *Proceedings of the 13th International Conference of the Learning Sciences*.

[34] Thomas M. Breuel. 2007. The hOCR microformat for OCR workflow and results. In *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, Vol. 2. 1063–1067. https://doi.org/10.1109/ICDAR.2007.4377078

[35] Thomas M. Breuel. 2008. The OCRopus open source OCR system. In *International Society for Optics and Photonics*, Berrin A. Yanikoglu and Kathrin Berkner (Eds.). 68150F–68150F–15. https://doi.org/10.1117/12.783598

[36] Dung C Bui, Joel Myerson, Sandra Hale, Dung C Bui, Joel Myerson, and Sandra Hale. 2012. Exploring Alternative Strategies for Improved Recall Note-Taking With Computers: Exploring Alternative Strategies for Improved Recall. *Journal of Educational Psychology Note-Taking With Computers* (2012). https://doi.org/10.1037/a0030367

[37] Patricia Burch, Annalee Good, and Carolyn Heinrich. 2016. Improving Access to, Quality , and the Effectiveness of Digital Tutoring in K 12 Education. *Educational Evaluation and Policy Analysis* 38, 1 (2016), 65–87. https://doi.org/10.3102/0162373715592706

[38] Georg Buscher. 2010. Eye Tracking Analysis of Preferred Reading Regions on the Screen. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*. ACM, Atlanta, 3307–3312.

[39] Georg Buscher, Edward Cutrell, and Meredith Ringel Morris. 2009. What Do You See When You're Surfing? Using Eye Tracking to Predict Salient Regions of Web Pages. In *CHI '09 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 21–30. https://doi.org/10.1145/1518701.1518705

[40] Georg Buscher, Andreas Dengel, Ludger Van Elst, and Florian Mittag. 2008. Generating and using gaze-based document annotations. In *CHI 2008 extended abstracts on Human factors in computing systems*. 3045–3050. https://doi.org/10.1145/1358628.1358805

[41] Georg Buscher, Ludger Van Elst, and Andreas Dengel. 2009. Segment-level display time as implicit feedback: A comparison to eye tracking. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09* (2009), 67–74. https://doi.org/10.1145/1571941.1571955

[42] Georg Buscher, Ryen W White, and Susan T Dumais. 2012. Large-Scale Analysis of Individual and Task Differences in Search Result Page Examination Strategies. In *Proceedings of the fifth ACM international conference on Web search and data mining*. 373–382.

[43] Gaston R. Cangiano and James D. Hollan. 2009. Capturing and restoring the context of everyday work: A case study at a law office. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5619 LNCS (2009), 945–954. https://doi.org/10.1007/978-3-642-02806-9_108

[44] Nicholas Carr. 2008. Is Google Making Us Stupid? *The Atlantic* 107, 2 (2008), 89–94. https://doi.org/10.1111/j.1744-7984.2008.00172.x

[45] Nicholas G. Carr. 2011. *The shallows: what the Internet is doing to our brains*. 280 pages.

[46] Jason Carter and Prasun Dewan. 2010. Design, implementation, and evaluation of an approach for determining when programmers are having difficulty. In *Proceedings of the 16th ACM international conference on Supporting group work - GROUP '10*. ACM Press, New York, New York, USA, 215–224. https://doi.org/10.1145/1880071.1880109

[47] Sathena Chan. 2017. Using keystroke logging to understand writers' processes on a reading-into-writing test. *Language Testing in Asia* 7, 1 (2017). https://doi.org/10.1186/s40468-017-0040-5

[48] Tsung-Hsiang Chang, Tom Yeh, and Robert C. Miller. 2010. GUI testing using computer vision. *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10* Figure 1 (2010), 1535. https://doi.org/10.1145/1753326.1753555

[49] Roger Chartier and Eric D. Friedman. 1997. The End of the Reign of the Book. *SubStance* 26, 1 (1997), 9. https://doi.org/10.2307/3684828

[50] Mon-chu Chen, John R. Anderson, and Myeong-Ho Sohn. 2001. What can a mouse cursor tell us more? Correlation of eye/mouse movements on web browsing Mon-Chu. In *CHI'01 extended abstracts on Human factors in computing systems*. 281–282. https://doi.org/10.1145/634067.634234

[51] Shiwei Cheng, Zhiqiang Sun, Lingyun Sun, Kirsten Yee, and Anind K Dey. 2015. Gaze-Based Annotations for Reading Comprehension. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, Vol. 1. 1569–1572. https://doi.org/10.1145/2702123.2702271

[52] Ed H. Chi, Lichan Hong, Michelle Gumbrecht, and Stuart K. Card. 2005. ScentHighlights. *Proceedings of the 10th international conference on Intelligent user interfaces - IUI '05* May 2014 (2005), 272. https://doi.org/10.1145/1040830.1040895

[53] Chinmay Chinara, Nishant Nath, Subhajeet Mishra, Sangram K Sahoo, and Farida A Ali. 2012. A novel approach to skew-detection and correction of English alphabets for OCR. In *IEEE Student Conference on Research and Development*. IEEE, 241–244. http://arxiv.org/abs/1801.00824

[54] Mark Claypool, Phong Le, Makoto Wased, and David Brown. 2001. Implicit interest indicators. *Proceedings of the 6th international conference on Intelligent user interfaces - IUI '01* (2001), 33–40. https://doi.org/10.1145/359784.359836 arXiv:31

[55] Andy Cockburn, Carl Gutwin, and Jason Alexander. 2006. Faster document navigation with space-filling thumbnails. *Proceedings of the SIGCHI '06* (2006), 1–10. https://doi.org/10.1145/1124772.1124774

[56] Jennifer G Cromley and Roger Azevedo. 2009. Locating Information within Extended Hypermedia. *Educational Technology Research and Development* 57, 3 (2009), 287–313. http://www.jstor.org/stable/40388631

[57] Andrew Cross, Mydhili Bayyapunedi, Dilip Ravindran, Edward Cutrell, and William Thies. 2014. VIdwiki: Enabling the Crowd to Improve the Legibility of Online Educational Videos. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing - CSCW '14*. 1167–1175. https://doi.org/10.1145/2531602.2531670

[58] Sarah D'angelo and Andrew Begel. 2017. Improving Communication Between Pair Programmers Using Shared Gaze Awareness. In *Chi*. https://doi.org/10.1145/3025453.3025573

[59] Pablo Delgado, Cristina Vargas, Rakefet Ackerman, and Ladislao Salmerón. 2018. Don't throw away your printed books: A meta-analysis on the effects of reading media on reading comprehension. *Educational Research Review* (sep 2018), 1–39. https://doi.org/10.1016/j.edurev.2018.09.003

[60] Morgan Dixon, James Fogarty, and Jacob Wobbrock. 2012. A general-purpose target-aware pointing enhancement using pixel-level analysis of graphical interfaces. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. ACM Press, New York, New York, USA, 3167. https://doi.org/10.1145/2207676.2208734

[61] Morgan Dixon, Alexander Nied, and James Fogarty. 2014. Prefab Layers and Prefab Annotations: Extensible Pixel-based Interpretation of Graphical Interfaces. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (2014), 221–230. https://doi.org/10.1145/2642918.2647412

[62] Morgan E Dixon, Gierad Laput, and James A Fogarty. 2014. Pixel-based Methods for Widget State and Style in a Runtime Implementation of Sliding Widgets. *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems* (2014), 2231–2240. https://doi.org/10.1145/2556288.2556979

[63] Education Super Highway. 2018. *Expanding digital learning to every classroom, every day*. Technical Report October. https://s3-us-west-1.amazonaws.com/esh-sots-pdfs/2018StateoftheStates.pdf

[64] John Evershed and Kent Fitch. 2014. Correcting noisy OCR: context beats confusion. In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage - DATeCH '14*. 45–51. https://doi.org/10.1145/2595188.2595200

[65] C Fabian Benitez-Quiroz, Ramprakash Srinivasan, and Aleix M Martinez. 2016. EmotioNet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 5562–5570. https://doi.org/10.1109/CVPR.2016.600

[66] Henry Feild, James Allan, and Rosie Jones. 2010. Predicting Searcher Frustration. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 34–41.

[67] Alejandro Figueroa. 2015. Exploring effective features for recognizing the user intent behind web queries. *Computers in Industry* 68 (apr 2015), 162–169. https://doi.org/10.1016/j.compind.2015.01.005

[68] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. 2005. Evaluating Implicit Measures to Improve Web Search. *ACM Transactions on Information Systems (TOIS)* 23, 2 (2005), 147–168.

[69] Juliane Franze, Kim Marriott, and Michael Wybrow. 2015. Does a Split-View Aid Navigation Within Academic Documents?. In *Proceedings of the 2015 ACM Symposium on Document Engineering - DocEng '15*. ACM Press, New York, New York, USA, 211–214. https://doi.org/10.1145/2682571.2797093

[70] Luanne Freund, Rick Kopak, and Heather O'Brien. 2016. The effects of textual environment on reading comprehension: Implications for searching as learning. *Journal of Information Science* 42, 1 (feb 2016), 79–93. https://doi.org/10.1177/0165551515614472

[71] Joel Friedlander. 2015. Updating Your Ebook After Publication. (2015). https://www.thebookdesigner.com/2015/05/updating-your-ebook-after-publication/

[72] Christian Frisson, Sylvain Malacria, Gilles Bailly, and Thierry Dutoit. 2016. InspectorWidget: a System to Analyze Users Behaviors in Their Applications. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*. 1548–1554. https://doi.org/10.1145/2851581.2892388

[73] Takugo Y Fukaya, Susumu Ono, Minoru Minakuchi, Seiya Nakashima, Masako Hayashi, and Hiroshi Ando. 2011. Reading text on a smart phone: Scrolling vs. paging: Toward designing effective electronic manuals. In *2011 International Conference on User Science and Engineering (i-USEr)*. IEEE, 59–63. https://doi.org/10.1109/iUSEr.2011.6150537

[74] Velayathan Ganesan and Seiji Yamada. 2007. Can We Find Common Rules of Browsing Behavior?. In *16th International Conference on World Wide Web*.

[75] Marelys L Garcia and Cesar I Caldera. 1996. The effect of color and typeface on the readability of on-line text. *Computers & Industrial Engineering* 31, 1-2 (1996), 519 –524. https://doi.org/10.1016/0360-8352(96)00189-1

[76] Dan B. Goldman, Chris Gonterman, Brian Curless, David Salesin, and Steven M. Seitz. 2008. Video object annotation, navigation, and composition. *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08* (2008), 3. https://doi.org/10.1145/1449715.1449719

[77] Susan Goodwin. 2005. Using screen capture software for web site usability and redesign buyin. *Library Hi Tech* 23, 4 (2005), 610–621. https://doi.org/10.1108/07378830510636382

[78] Qi Guo and Eugene Agichtein. 2008. Exploring Mouse Movements for Inferring Query Intent Categories and Subject Descriptors. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 707–708.

[79] Qi Guo and Eugene Agichtein. 2010. Towards Predicting Web Searcher Gaze Position from Mouse Movements. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*. 3601–3606. https://doi.org/10.1145/1753846.1754025

[80] Qi Guo and Eugene Agichtein. 2012. Beyond dwell time: Estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 569–578. https://doi.org/10.1145/2187836.2187914

[81] Qi Guo, Haojian Jin, Dmitry Lagun, Shuai Yuan, and Eugene Agichtein. 2013. Mining touch interaction data on mobile devices to predict web search result relevance. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '13* (2013), 153. https://doi.org/10.1145/2484028.2484100

[82] Qi Guo, One Microsoft Way, and Shuai Yuan. 2013. Towards estimating web search result relevance from touch interactions on mobile devices. *CHI'13 Extended Abstracts ...* (2013), 1821–1826. https://doi.org/10.1145/2468356.2468683

[83] Richard H. Hall and Patrick Hanna. 2004. The impact of web page text-background colour combinations on readability, retention, aesthetics and behavioural intention. *Behaviour and Information Technology* 23, 3 (2004), 183–195. https://doi.org/10.1002/9781118534168.ch13

[84] Shuguang Han, I. Han Hsiao, and Denis Parra. 2014. A study of mobile information exploration with multi-touch interactions. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8393 LNCS (2014), 269–276. https://doi.org/10.1007/978-3-319-05579-4_33

[85] Shuguang Han, Zhen Yue, and Daqing He. 2015. Understanding and Supporting Cross-Device Web Search for Exploratory Tasks with Mobile Touch Interactions. *ACM Transactions on Information Systems* 33, 4 (2015), 1–34. https://doi.org/10.1145/2738036

[86] Johannes Harms, Martina Kratky, Christoph Wimmer, Karin Kappel, and Thomas Grechenig. 2015. Navigation in long forms on smartphones: Scrolling worse than tabs, menus, and collapsible fieldsets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9298 (2015), 333–340. https://doi.org/10.1007/978-3-319-22698-9_21

[87] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. 2017. Going deeper into action recognition: A survey. *Image and Vision Computing* 60 (2017), 4–21. https://doi.org/10.1016/j.imavis.2017.01.010 arXiv:1605.04988

[88] Peter Hernon, Rosita Hopper, Michael R. Leach, Laura L. Saunders, and Jane Zhang. 2007. E-book Use by Students: Undergraduates in Economics, Literature, and Nursing. *Journal of Academic Librarianship* 33, 1 (2007), 3–13. https://doi.org/10.1016/j.acalib.2006.08.005

[89] Daniel Hienert and Dagmar Kern. 2017. Term-Mouse-Fixations as an Additional Indicator for Topical User Interests in Domain-Specific Search. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval - ICTIR '17*. ACM Press, New York, New York, USA, 249–252. https://doi.org/10.1145/3121050.3121088

[90] Terje Hillesund. 2010. Digital reading spaces: How expert readers handle books, the Web and electronic paper. *First Monday* 15 (apr 2010), 1–19. https://doi.org/10.5210/fm.v15i4.2762

[91] Eric Horvitz, Jack Breese, David Heckerman, David Hovel, Koos Rommelset, and Koos Rommelse. 1998. The Lumiere Project: Bayesian User Modeling for Inferring

the Goals and Needs of Software Users. *Fourteenth Conference on Uncertainty in Artificial Intelligence* (1998), 256–265. arXiv:1301.7385

[92] Jeff Huang and Abdigani Diriye. 2012. Web User Interaction Mining from Touch-Enabled Mobile Devices. In *HCIR workshop*. https://sites.google.com/site/hcirworkshop/hcir-2012/posters

[93] Jeff Huang, Ryen W White, and Susan Dumais. 2011. No clicks, no problem: Using cursor movements to understand and improve search. In *Proceedings of the 29th SIGCHI Conference on Human Factors in Computing Systems*. 1225. https://doi.org/10.1145/1978942.1979125

[94] Joan Figuerola Hurtado. 2016. Attention Span For Personalisation. (jul 2016), 1–6. arXiv:1608.00147 http://arxiv.org/abs/1608.00147

[95] Hypothes.is. 2011. Hypothes.is. (2011). http://hypothes.is

[96] Evdokiya D Ignatova and Willem-paul Brinkman. 2007. Clever Tracking User Behaviour over the Web: Enabling Researchers to Respect the User. *BCS-HCI '07 Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...but not as we know it - Volume 2* 2, September (2007), 179–182. https://doi.org/978-1-902505-95-4

[97] Bonnie Imler and Michelle Eichelberger. 2011. Using screen capture to study user research behavior. *Library Hi Tech* 29, 3 (2011), 446–454. https://doi.org/10.1108/07378831111174413

[98] IMotions A/S. 2016. iMotions Biometric Research Platform 6.3. (2016).

[99] Shamsi T. Iqbal and Eric Horvitz. 2007. Disruption and recovery of computing tasks. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07* (2007), 677. https://doi.org/10.1145/1240624.1240730

[100] Noman Islam, Zeeshan Islam, and Nazia Noor. 2016. A Survey on Optical Character Recognition System. *Journal of Information & Communication Technology-JICT* 10, 2 (2016), 1–4. arXiv:1710.05703 https://arxiv.org/pdf/1710.05703.pdf

[101] Ferris Jabr. 2013. Why the Brain Prefers Paper Scientific American, November 2013. November (2013), 48–53.

[102] Mark Jensen and Lauren Scharff. 2014. Using E-Book annotations to develop deep reading. *Journal of Teaching and Learning with Technology* 3, 2 (2014), 83. https://doi.org/10.14434/jotlt.v3n2.12872

[103] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. 2012. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 7 (2012), 1409–1422. https://doi.org/10.1109/TPAMI.2011.239 arXiv:arXiv:1412.7522v1

[104] Zekeriya Kazanci. 2015. University Students' Preferences of Reading from a Printed Paper or a Digital Screen A Longitudinal Study. *International Journal of Culture and History (EJournal)* 1, 1 (2015), 50–53. https://doi.org/10.18178/ijch.2015.1.1.009

[105] Diane Kelly and Nicholas J. Belkin. 2001. Reading time, scrolling and interaction: exploring implicit sources of user preferences for relevance feedback. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* 558 (2001), 408–409. https://doi.org/10.1145/383952.384045

[106] Diane Kelly and Nicholas J Belkin. 2004. Display Time as Implicit Feedback : Understanding Task Effects. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '04* (2004), 377–384. https://doi.org/10.1145/1008992.1009057

[107] Do-Hong Kim and Huynh Huynh. 2008. Computer-based and paper-and-pencil administration mode effects on a statewide end-of-course English test. *Educational and Psychological Measurement* 68, 4 (2008), 554–570. https://doi.org/10.1177/0013164407310132

[108] Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J. Guo, Robert C. Miller, and Krzysztof Z. Gajos. 2014. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14* (2014), 4017–4026. https://doi.org/10.1145/2556288.2556986

[109] Jaewon Kim, Paul Thomas, Ramesh Sankaranarayana, Tom Gedeon, and Hwan-Jin Yoon. 2015. Eye-tracking analysis of user behavior and performance in web search on large and small screens. *Journal of the Association for Information Science and Technology* 66, 3 (mar 2015), 526–544. https://doi.org/10.1002/asi.23187

[110] Youngho Kim, Ahmed Hassan, Ryen W. White, and Imed Zitouni. 2014. Modeling dwell time to predict click-level satisfaction. *Proceedings of the 7th ACM international conference on Web search and data mining - WSDM '14* (2014), 193–202. https://doi.org/10.1145/2556195.2556220

[111] Mark King. 2012. Amazon wipes customer's Kindle and deletes account with no explanation. (oct 2012). https://www.theguardian.com/money/2012/oct/22/amazon-wipes-customers-kindle-deletes-account

[112] Charles K Kinzer, Selen Turkay, Daniel L Hoffman, Nilgun Gunbas, Pantiphar Chantes, Apichai Chaiwinij, and Tatyana Dvorkin. 2012. Examining the Effects of Text and Images on Story Comprehension: An Eye-Tracking Study of Reading in a Video Game and Comic Book. *61st Yearbook of the Literacy Research Association* 259 (2012), 259–275.

[113] Zuzanna Kłyszejko, Tomasz Soluch, Anna Wieczorek, Karolina Chmiel, Justyna Sarzyńska, and Agnieszka Szóstek. 2011. The influence of text visualization on the screen on eye movements and information processing. In *Proceedings of the Conference: Interfejs użytkownika - Kansei w praktyce, Warszawa*. 113–120.

[114] Kenneth R Koedinger, Elizabeth a Mclaughlin, Jihee Kim, Julianna Zhuxin Jia, and Norman L Bier. 2015. Learning is Not a Spectator Sport: Doing is Better than Watching for Learning from a MOOC. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale - L@S '15*. 111–120. https://doi.org/10.1145/2724660.2724681

[115] Terttu Kortelainen. 2015. Reading Format Preferences of Finnish University Students. In *Information Literacy: Moving Toward Sustainability*, Anja Wintermeyer and Kathrin Knautz (Eds.). Communications in Computer and Information Science, 446–454. https://doi.org/10.1007/978-3-319-28197-1

[116] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye Tracking for Everyone. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2176–2184.

[117] Kai Kunze, Hitoshi Kawaichi, Kazuyo Yoshimura, and Koichi Kise. 2013. The wordometer - Estimating the number of words read using document image retrieval and mobile eye tracking. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR* (2013), 25–29. https://doi.org/10.1109/ICDAR.2013.14

[118] Kai Kunze, Hitoshi Kawaichi, Kazuyo Yoshimura, and Koichi Kise. 2013. Towards inferring language expertise using eye tracking. In *Chi*. 217. https://doi.org/10.1145/2468356.2468396

[119] Kai Kunze, Yuzuko Utsumi, Yuki Shiga, Koichi Kise, and Andreas Bulling. 2013. I know what you are reading. In *Proceedings of the 17th annual international symposium on International symposium on wearable computers - ISWC '13*. 113. https://doi.org/10.1145/2493988.2494354

[120] Dmitry Lagun, Mikhail Ageev, Qi Guo, and Eugene Agichtein. 2014. Discovering common motifs in cursor movement data for improving web search. *Proceedings of the 7th ACM international conference on Web search and data mining - WSDM '14* (2014), 183–192. https://doi.org/10.1145/2556195.2556265

[121] Dmitry Lagun, Chih-Hung Hsieh, Dale Webster, and Vidhya Navalpakkam. 2014. Towards better measurement of attention and satisfaction in mobile search. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval - SIGIR '14*. ACM Press, New York, New York, USA, 113–122. https://doi.org/10.1145/2600428.2609631

[122] Dmitry Lagun and Mounia Lalmas. 2016. Understanding and Measuring User Engagement and Attention in Online News Reading. *Proceedings of the Ninth ACM*

International Conference on Web Search and Data Mining - WSDM '16 (2016), 113–122. https://doi.org/10.1145/2835776.2835833

[123] Fabrizio Lamberti and Gianluca Paravati. 2015. VDHM: Viewport-DOM Based Heat Maps as a Tool for Visually Aggregating Web Users' Interaction Data from Mobile and Heterogeneous Devices. In *2015 IEEE International Conference on Mobile Services*, Vol. 32. IEEE, 33–40. https://doi.org/10.1109/MobServ.2015.15

[124] Fabrizio Lamberti, Gianluca Paravati, Valentina Gatteschi, and Alberto Cannavo. 2017. Supporting Web Analytics by Aggregating User Interaction Data From Heterogeneous Devices Using Viewport-DOM-Based Heat Maps. *IEEE Transactions on Industrial Informatics* 13, 4 (aug 2017), 1989–1999. https://doi.org/10.1109/TII.2017.2658663

[125] Ane Land, Angela Repanovici, and Almuth Gastinger. 2015. The More they Tried it the Less they Liked it : Norwegian and Romanian Student ' s Response to Electronic Course Material. In *Information Literacy: Moving Toward Sustainability*, Anja Wintermeyer and Kathrin Knautz (Eds.), Vol. 1. Communications in Computer and Information Science, 455–463. https://doi.org/10.1007/978-3-319-28197-1

[126] Colin Lankshear and Michele Knobel. 2013. *A new literacies reader: Educational perspectives*. Peter Lang International Academic Publishers. 5 pages. https://dl.acm.org/citation.cfm?id=2556166

[127] Walter S. Lasecki, Mitchell Gordon, Danai Koutra, Malte F. Jung, Steven P. Dow, and Jeffrey P. Bigham. 2014. Glance: Rapidly Coding Behavioral Video with the Crowd. *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14* (2014), 551–562. https://doi.org/10.1145/2642918.2647367

[128] Mariëlle Leijten and Luuk Van Waes. 2006. Inputlog : A logging tool for the research of writing processes. (2006).

[129] Juho Leinonen. 2017. Preventing Keystroke Based Identification in Open Data Sets. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*. 101–109.

[130] Luis A Leiva and Roberto Vivó. 2012. Interactive Hypervideo Visualization for Browsing Behavior Analysis. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 381–384.

[131] Fwa Hua Leong. 2016. Fine-Grained Detection of Programming Students' Frustration Using Keystrokes, Mouse Clicks and Interaction Logs. *Open Journal of Social Sciences* 04, 09 (2016), 9–18. https://doi.org/10.4236/jss.2016.49002

[132] Liang Yi Li, Gwo Dong Chen, and Sheng Jie Yang. 2013. Construction of cognitive maps to improve e-book reading and navigation. *Computers and Education* 60, 1 (2013), 32–39. https://doi.org/10.1016/j.compedu.2012.07.010

[133] Yixuan Li, Pingmei Xu, Dmitry Lagun, and Vidhya Navalpakkam. 2017. Towards Measuring and Inferring User Interest from Gaze. *WWW (Companion Volume)* (2017), 525–533. https://doi.org/10.1145/3041021.3054182

[134] Junchi Liang and Abdeslam Boularias. 2018. Task-Relevant Object Discovery and Categorization for Playing First-person Shooter Games. *arXiv preprint arXiv:1806.06392* (jun 2018), 1–10. arXiv:1806.06392 http://arxiv.org/abs/1806.06392

[135] Daniel J. Liebling and Susan T Dumais. 2014. Gaze and Mouse Coordination in Everyday Work. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. 1141–1150. https://doi.org/10.1145/2638728.2641692

[136] Jens O Liegle and Thomas N Janicki. 2006. The effect of learning styles on the navigation needs of Web-based learners. *Computers in Human Behavior* 22 (2006), 885–898. https://doi.org/10.1016/j.chb.2004.03.024

[137] Keng Pei Lin, Chia Yu Lai, Po Cheng Chen, and San Yih Hwang. 2016. Personalized Hotel Recommendation Using Text Mining and Mobile Browsing Tracking. *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015* (2016), 191–196. https://doi.org/10.1109/SMC.2015.46

[138] Chao Liu, Ryen W. White, and Susan Dumais. 2010. Understanding web browsing behaviors through Weibull analysis of dwell time. *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10* (2010), 379. https://doi.org/10.1145/1835449.1835513

[139] Y. Liu, B. Gao, T.Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li. 2008. Browserank: Letting web users vote for page importance. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (2008), 451–458. https://doi.org/10.1145/1390334.1390412

[140] Ziming Liu. 2005. Reading behavior in the digital environment. *Journal of Documentation* 61, 6 (dec 2005), 700–712. https://doi.org/10.1108/00220410510632040

[141] Steven G Luke, John M Henderson, and Fernanda Ferreira. 2015. Children's Eye-Movements During Reading Reflect the Quality of Lexical Representations: An Individual Differences Approach. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 41, 6 (2015), 1675–1683. https://doi.org/10.1037/xlm0000133

[142] Laura MacLeod, Margaret-Anne Storey, and Andreas Bergen. 2015. Code, Camera, Action: How Software Developers Document and Share Program Knowledge Using YouTube. In *2015 IEEE 23rd International Conference on Program Comprehension*, Vol. 2015-Augus. IEEE, 104–114. https://doi.org/10.1109/ICPC.2015.19

[143] Anne Mangen and Don Kuiken. 2014. Lost in an iPad: Narrative engagement on paper and tablet. *Scientific Study of Literature* 4, 2 (2014), 150–177. https://doi.org/10.1075/ssol.4.2.02man

[144] Anne Mangen and Adriaan van der Weel. 2016. The evolution of reading in the age of digitisation: an integrative framework for reading research. *Literacy* 50, 3 (sep 2016), 116–124. https://doi.org/10.1111/lit.12086

[145] Sara J. Margolin, Casey Driscoll, Michael J. Toland, and Jennifer Little Kegler. 2013. E-readers, computer screens, or paper: Does reading comprehension change across media platforms? *Applied Cognitive Psychology* 27, 4 (2013), 512–519. https://doi.org/10.1002/acp.2930

[146] Gloria Mark, Victor M. Gonzalez, and Justin Harris. 2005. No task left behind? *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '05* (2005), 321. https://doi.org/10.1145/1054972.1055017

[147] Catherine Marshall. 1997. Annotation: from paper books to the digital library. *Proceedings of the second ACM international conference on Digital libraries - DL '97* (1997), 131–140. https://doi.org/10.1145/263690.263806

[148] Pascual Martínez-Gómez, Tadayoshi Hara, and Akiko Aizawa. 2012. Recognizing Personal Characteristics of Readers using Eye-Movements and Text Features. *Proceedings of COLING 2012* December (2012), 1747–1762. http://www.aclweb.org/anthology/C12-1107

[149] MK McGowan, PR Stephen, and CW Bradley. 2009. Student perceptions of electronic textbooks. *Issues in Information Systems* 10, 2 (2009), 459–465. http://www.iacis.org/iis/2009/P2009

[150] Jim Morey, John Gammack, and Erik S. Thornquist. 2015. Interface development for a gaze-controlled reading support application. *2015 International Conference on Information and Communication Technology Research, ICTRC 2015* (2015), 214–217. https://doi.org/10.1109/ICTRC.2015.7156460

[151] M Morita and Y Shinoda. 1994. Information filtering based on user behavior analysis and best match text retrieval. *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (1994), 272–281. https://doi.org/citation.cfm?id=188490.188583

[152] Sanaz Motamedi and Pilsung Choe. 2015. Smartphone Information Displays When Reading News in Persian and English Languages. *International Journal of Human-Computer Interaction* 31, 6 (2015), 427–439. https://doi.org/10.1080/10447318.2015.1038129

[153] Florian Mueller and Andrea Lockerd. 2001. Cheese: tracking mouse movement activity on websites, a tool for user modeling. *CHI'01 extended abstracts on Human factors in ...* (2001), 279–280. https://doi.org/10.1145/634067.634233

[154] Pam A Mueller and Daniel M Oppenheimer. 2014. The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking. *Psychological Science* 25, 6 (2014), 1159–1168. https://doi.org/10.1177/0956797614524581

[155] Cuong Nguyen and Feng Liu. 2015. Making Software Tutorial Video Responsive. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* (2015), 1565–1568. https://doi.org/10.1145/2702123.2702209

[156] Kenton O'Hara and Abigail Sellen. 1997. A Comparison of Reading Paper and On-Line Documents. In *Proceedings of CHI '97, Human Factors in Computing Systems*. Atlanta, Georgia, U.S.A., 335–342. https://doi.org/10.1145/258549.258787

[157] Kieron O'Hara, Mischa M. Tuffield, and Nigel Shadbolt. 2008. Lifelogging: Privacy and empowerment with memories for life. *Identity in the Information Society* 1, 1 (2008), 155–172. https://doi.org/10.1007/s12394-009-0008-4

[158] Nuria Oliver, Greg Smith, Chintan Thakkar, and Arun C Surendran. 2006. SWISH: semantic analysis of window titles and switching history. In *IUI '06: Proceedings of the 11th international conference on Intelligent User Interfaces*. 194–201. https://doi.org/10.1145/1111449.1111492

[159] Takayuki Omori, Shinpei Hayashi, Katsuhisa Maruyama, and Takayuki Omori. 2015. A Survey on Methods of Recording Fine-grained Operations on Integrated Development Environments and their Applications. *Computer Software* 32, 1 (2015), 1_60–1_60.

[160] Kevin Ong, Kal Jarvelin, Falk Scholer, and Mark Sanderson. 2018. QWERTY: The Effects of Typing on Web Search Behavior. In *Proceedings of ACM SIGIR Conference on Human Information Interaction and Retrieval*. http://marksanderson.org/publications/my

[161] Alexandra Papoutsaki, Patsorn Sangkloy, James Laskey, Nediyana Daskalova, Jeff Huang, and James Hays. 2016. WebGazer: Scalable Webcam Eye Tracking Using User Interactions. In *IJCAI*. 3839–3845.

[162] Ravi Parikh. 2013. Distribution of Word Lengths in Various Languages. (2013). http://www.ravi.io/language-word-lengths

[163] Jungkook Park, Yeong Hoon Park, Suin Kim, and Alice Oh. 2017. Eliph: Effective Visualization of Code History for Peer Assessment in Programming Education. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. 458–467. https://doi.org/10.1145/2998181.2998285

[164] Marc Parry. 2013. Students Get Savvier About Textbook Buying. (2013), 8 pages. http://chronicle.com/article/Students-Get-Savvier-About/136827/

[165] Kevin B Paterson, Victoria A Mcgowan, Timothy R Jordan, and Kevin Paterson. 2015. Children and adults both see pirates' in parties': letter-position effects for

developing readers and skilled adult readers. *Developmental Science* 18, 2 (2015), 335–343.

[166] Manuel Perea and Stephen J Lupker. 2003. Transposed-Letter Confusability Effects in Masked Form Priming. In *Masked priming: State of the art*. 97–120.

[167] Annie Piolat, Jean-Yves Roussey, and Olivier Thunin. 1997. Effects of screen presentation on text reading and revising. *International Journal of Human-Computer Studies* 47, 4 (1997), 565–589. https://doi.org/10.1006/ijhc.1997.0145

[168] Hector R. Ponce and Richard E. Mayer. 2014. Qualitatively different cognitive processing during online reading primed by different study activities. *Computers in Human Behavior* 30 (2014), 121–130. https://doi.org/10.1016/j.chb.2013.07.054

[169] Luca Ponzanelli, Gabriele Bavota, Andrea Mocci, Massimiliano Di Penta, Rocco Oliveto, Mir Hasan, Barbara Russo, Sonia Haiduc, and Michele Lanza. 2016. Too long; didn't watch!: extracting relevant fragments from software development video tutorials. *Proceedings of the 38th International Conference on Software Engineering - ICSE '16* (2016), 261–272. https://doi.org/10.1145/2884781.2884824

[170] Luca Ponzanelli, Gabriele Bavota, Andrea Mocci, Rocco Oliveto, Massimiliano Di Penta, Sonia Cristina Haiduc, Barbara Russo, and Michele Lanza. 2017. Automatic Identification and Classification of Software Development Video Tutorial Fragments. *IEEE Transactions on Software Engineering* 14, 8 (2017). https://doi.org/10.1109/TSE.2017.2779479

[171] Luca Ponzanelli, Gabriele Bavota, Andrea Mocci, Massimiliano Di Penta, Rocco Oliveto, Barbara Russo, Sonia Haiduc, and Michele Lanza. 2016. CodeTube: Extracting Relevant Fragments from Software Development Video Tutorials. In *IEEE/ACM International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 645–648.

[172] Deepak Prasad and Tsuyoshi Usagawa. 2014. Scoping the Possibilities : Student Preferences towards Open Textbooks Adoption for E-Learning. *Creative Education* 5, 24 (2014), 2027–2040. https://doi.org/10.4236/ce.2014.524227

[173] Eric Price, Jennifer 8. Lee, and Greg Price. 2012. NewsDiffs: Tracking Online News Over Time. (2012). http://www.newsdiffs.org

[174] Maja Pusara and Carla E Brodley. 2004. User Re-Authentication via Mouse Movements. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*. ACM, 1–8.

[175] Mohamed Zaki Ramadan. 2011. Evaluating college students' performance of Arabic typeface style, font size, page layout and foreground/background color combinations of e-book materials. *Journal of King Saud University - Engineering Sciences* 23, 2 (2011), 89–100. https://doi.org/10.1016/j.jksues.2011.03.005

[176] Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin* 124, 3 (1998), 372–422. https://doi.org/10.1037/0033-2909.124.3.372 arXiv:hep-th/0010225v1

[177] Keith Rayner, Sarah J. White, Rebecca L. Johnson, and Simon P. Liversedge. 2006. Raeding wrods with jubmled letetrs: There is a cost. *Psychological Science* 17 (2006), 192–193.

[178] Sam Roberts. 2005. 1920 : Women Get the Vote. (2005). https://www.nationsreportcard.gov/reading

[179] Amanda J. Rockinson-Szapkiw, Jennifer Courduff, Kimberly Carter, and David Bennett. 2013. Electronic versus traditional print textbooks: A comparison study on the influence of university students' learning. *Computers & Education* 63 (apr 2013), 259–266. https://doi.org/10.1016/j.compedu.2012.11.022

[180] Kerry Rodden, Mountain View, Xin Fu, Anne Aula, and Ian Spiro. 2008. Eye-Mouse Coordination Patterns on Web Search Results Pages. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*. 2997–3002. https://doi.org/10.1145/1358628.1358797

[181] Ernst Z. Rothkopf. 1971. Incidental memory for location of information in text. *Journal of Verbal Learning and Verbal Behavior* 10, 6 (1971), 608–613. https://doi.org/10.1016/S0022-5371(71)80066-X

[182] Adam Rule. 2015. Visualizing Computer Activity to Support the Resumption of Long-term Creative Work. *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition - C&C '15* (2015), 343–344. https://doi.org/10.1145/2757226.2764772

[183] Madjid Sadallah, Benoît Encelle, Azze Eddine Maredj, and Yannick Prié. 2015. Towards reading session-based indicators in educational reading analytics. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9307 (2015), 297–310. https://doi.org/10.1007/978-3-319-24258-3_22

[184] Paul Saenger. 1997. *Space Between Words: The Origins of Silent Reading*. Stanford University Press. https://doi.org/DrakeZ1003.S131997(NOS)

[185] Christopher a. Sanchez and Jennifer Wiley. 2009. To Scroll or Not to Scroll: Scrolling, Working Memory Capacity, and Comprehending Complex Texts. *Human Factors* 51, 5 (2009), 730–738. https://doi.org/10.1177/0018720809352788

[186] Susan K. Schnipke and Marc W. Todd. 2000. Trials and tribulations of using an eye-tracking system. In *CHI '00 extended abstracts on Human factors in computing systems*. 273–274. https://doi.org/10.1145/633451.633452

[187] Sofie Schoonbaert and Jonathan Grainger. 2004. Letter position coding in printed word perception : Effects of repeated and transposed letters. *LANGUAGE AND COGNITIVE PROCESSES* 19, 3 (2004), 333–367. https://doi.org/10.1080/01690960344000198

[188] Klaus U Schulz and Stoyan Mihov. 2002. Fast string correction with Levenshtein automata. *International Journal on Document Analysis and Recognition* 5, 1 (2002), 67–85. https://doi.org/10.1007/s10032-002-0082-8

[189] Faisal Shafait. 2009. Document image analysis with OCRopus. In *2009 IEEE 13th International Multitopic Conference*. IEEE, 1–6. https://doi.org/10.1109/INMIC.2009.5383078

[190] Hijung Valentina Shin, Floraine Berthouzoz, Adobe Research, Wilmot Li, Frédo Durand, and Mit Csail. 2015. Visual Transcripts: Lecture Notes from Blackboard-Style Lecture Videos. *ACM Trans. Graph. Article* 34, 10 (2015), 2–5. https://doi.org/10.1145/2816795.2818123

[191] Gregory M. Shreve, Erik Angelone, and Isabel Lacruz. 2014. Efficacy of screen recording in the other-revision of translations: episodic memory and event models. *MonTI. Monografías de Traducción e Interpretación* (2014), 225–245. https://doi.org/10.6035/MonTI.2014.ne1.7

[192] Lauren M. Singer and Patricia A. Alexander. 2017. Reading Across Mediums: Effects of Reading Digital and Print Texts on Comprehension and Calibration. *Journal of Experimental Education* 85, 1 (2017), 155–172. https://doi.org/10.1080/00220973.2016.1143794

[193] Lauren M Singer and Patricia A Alexander. 2017. Reading on Paper and Digitally: What the Past Decades of Empirical Research Reveal. *Review of Educational Research* (2017), 1–35. https://doi.org/10.3102/0034654317722961

[194] Ray Smith. 2007. An Overview of the Tesseract OCR Engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2*, Vol. 2. IEEE, 629–633. https://doi.org/10.1109/ICDAR.2007.4376991 arXiv:arXiv:1011.1669v3

[195] Ray Smith. 2016. Tesseract Blends Old and New OCR Technology. In *International Workshop on Document Analysis Systems*. Santorini, Greece. https://github.com/tesseract-ocr/docs/blob/master/das

[196] Speedtest by Ookla. 2018. *Mobile Speedtest Data*. Technical Report June. https://www.speedtest.net/reports/united-states/

[197] Peter Stallybrass. 2002. Books and Scrolls: Navigating the Bible. In *Books and Readers in Early Modern England: Material Studies*, Jennifer Andersen and Elizabeth Sauer (Eds.). University of Pennsylvania Press, Chapter Books and, 42–79.

[198] Markus Strohmaier and Mark Kröll. 2012. Acquiring knowledge about human goals from Search Query Logs. *Information Processing and Management* 48, 1 (2012), 63–82. https://doi.org/10.1016/j.ipm.2011.03.010

[199] John C. Tang, Sophia B. Liu, Michael Muller, James Lin, and Clemens Drews. 2006. Unobtrusive but invasive: Using screen recording to collect field data on computer-mediated interaction. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work - CSCW '06*. ACM Press, New York, New York, USA, 479. https://doi.org/10.1145/1180875.1180948

[200] Mark Torrance, Roger Johansson, Victoria Johansson, and Åsa Wengelin. 2016. Reading during the composition of multi-sentence texts: an eye-movement study. *Psychological Research* 80, 5 (2016), 729–743.

[201] Paul Trapnell and Lisa Sinclair. 2013. Texting Frequency and The Moral Shallowing Hypothesis. In *Annual Meeting of the Society for Personality and Social Psychology*.

[202] Lana Traut and Janet Toland. 2014. Online vs . Printed Course Materials : Student Preferences and Influential factors. In *ICIS2014 SIGGreen Workshop*. https://siggreen.wikispaces.com/Accepted+Papers

[203] Hsiao-Yu Fish Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. 2017. Self-supervised Learning of Motion Capture. *Neural Information Processing Systems* (dec 2017), 1–11. arXiv:1712.01337 http://arxiv.org/abs/1712.01337

[204] Hans van der Meij and Jan van der Meij. 2016. Demonstration-based training (DBT) in the design of a video tutorial for software training. *Instructional Science* 44, 6 (2016), 527–542. https://doi.org/10.1007/s11251-016-9394-9

[205] Alessandro Vinciarelli and Jean-Marc Odobez. 2006. Application of Information Retrieval Technologies to Presentation Slides. *IEEE Transactions on Multimedia* 8, 5 (oct 2006), 981–995. https://doi.org/10.1109/TMM.2006.879870

[206] Gang Wang, Virginia Tech, Uc Santa Barbara XINYI ZHANG, Shiliang Tang, Uc Santa Barbara CHRISTO WILSON, Haitao Zheng, Ben Y Zhao, Xinyi Zhang, Christo Wilson, G Wang, X Zhang, S Tang, H Zheng, and B Y Zhao. 2017. Clickstream User Behavior Models. *ACM Trans. Web ACM Transactions on the Web* 0, 0 (2017). https://doi.org/10.1145/3068332

[207] Pingfei Wang and Qian Fei. 2013. Scrolling or Paging: The Impact of Interaction Style on the Search Result Page of Mobile Commerce Website. In *International Conference on Human-Computer Interaction*. 454–457. https://doi.org/10.1007/978-3-642-39476-8_92

[208] Erik Wästlund. 2007. *Experimental Studies of Human-Computer Interaction: Working memory and mental workload in complex cognition*. https://doi.org/10.1016/j.chb.2004.02.007

[209] Ulrich W Weger and Albrecht W Inhoff. 2007. Long-range regressions to previously read words are guided by spatial and verbal memory. *Memory & Cognition* 35, 6 (2007), 1293–1306.

[210] Sarah Weir, Juho Kim, Krzysztof Z Gajos, and Robert C Miller. 2015. Learnersourcing Subgoal Labels for How-to Videos. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing - CSCW '15*. ACM Press, New York, New York, USA, 405–416. https://doi.org/10.1145/2675133.2675219

[211] Mitchell Weisberg. 2011. Student attitudes and behaviors towards digital textbooks. *Publishing Research Quarterly* 27, 2 (2011), 188–196. https://doi.org/10.1007/s12109-011-9217-4

[212] Åsa Wengelin, Mark Torrance, Kenneth Holmqvist, Sol Simpson, David Galbraith, Victoria Johansson, and Roger Johansson. 2009. Combined eyetracking and keystroke-logging methods for studying cognitive processes. *Behavior Research Methods* 41, 2 (2009), 337–351. https://doi.org/10.3758/BRM.41.2.337

[213] Ryen W White and Georg Buscher. 2012. Text Selections As Implicit Relevance Feedback. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1151–1152. https://doi.org/10.1145/2348283.2348514

[214] Ryen W. White and Diane Kelly. 2006. A study on the effects of personalization and task information on implicit feedback performance. *Proceedings of the 15th ACM international conference on Information and knowledge management - CIKM '06* (2006), 297. https://doi.org/10.1145/1183614.1183659

[215] Wikipedia. 2017. ALTO (XML). (2017). https://en.wikipedia.org/wiki/ALTO

[216] Wikipedia. 2018. hOCR. (2018). https://en.wikipedia.org/wiki/HOCR

[217] Wikipedia. 2018. Tesseract (software). (2018). https://en.wikipedia.org/wiki/Tesseract

[218] Dan Wu. 2018. *Transforming Digital Worlds*. Lecture Notes in Computer Science, Vol. 10766. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-78105-1

[219] Haojin Yang, Christoph Oehlke, and Christoph Meinel. 2012. An automated analysis and indexing framework for lecture video portal. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7558 LNCS (2012), 285–294. https://doi.org/10.1007/978-3-642-33642-3_31

[220] Haojin Yang, Maria Siebert, Patrick Lühne, Harald Sack, and Christoph Meinel. 2011. Automatic lecture video indexing using video OCR technology. *Proceedings*

- *2011 IEEE InternationalSymposium on Multimedia, ISM 2011* (2011), 111–116. https://doi.org/10.1109/ISM.2011.26

[221] Haojin Yang, Maria Siebert, Patrick Luhne, Harald Sack, and Christoph Meinel. 2011. Lecture Video Indexing and Analysis Using Video OCR Technology. In *2011 Seventh International Conference on Signal Image Technology & Internet-Based Systems*, Vol. 2. IEEE, 54–61. https://doi.org/10.1109/SITIS.2011.20

[222] Xing Yi, Liangjie Hong, Erheng Zhong, Nathan Nan, and Liu Suju. 2014. Beyond Clicks : Dwell Time for Personalization. *Commun. ACM* (2014), 8. https://doi.org/10.1145/2505515.2505682

[223] Alper Yilmaz, Omar Javed, and Mubarak Shah. 2006. Object tracking. *Comput. Surveys* 38, 4 (2006), 13–es. https://doi.org/10.1145/1177352.1177355 arXiv:arXiv:1011.1669v3

[224] Vlasta Zabukovec and Polona Vilar. 2015. Paper or Electronic : Preferences of Slovenian Students. In *Information Literacy: Moving Toward Sustainability*, Anja Wintermeyer and Kathrin Knautz (Eds.). Communications in Computer and Information Science, 427–435. https://doi.org/10.1007/978-3-319-28197-1

[225] Saniya Zahoor, Mangesh Bedekar, and Pranali K Kosamkar. 2014. User Implicit Interest Indicators Learned from the Browser on the Client Side. *Proceedings of the 2014 International Conference on Information and Communication Technology for Competitive Strategies* (2014), 57:1—-57:4. https://doi.org/10.1145/2677855.2677912

[226] Saniya Zahoor, Mangesh Bedekar, and Varad Vishwarupe. 2016. A Framework to Infer Webpage Relevancy for a User. In *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1*, Suresh Chandra Satapathy and Swagatam Das (Eds.). Smart Innovation, Systems and Technologies, Vol. 50. Springer International Publishing, Cham, 147–154. https://doi.org/10.1007/978-3-319-30933-0_16

[227] Saniya Zahoor, Digvijaysingh Rajput, Mangesh Bedekar, and Pranali Kosamkar. 2015. Capturing, understanding and interpreting user interactions with the browser as implicit interest indicators. In *2015 International Conference on Pervasive Computing (ICPC)*. IEEE, 1–6. https://doi.org/10.1109/PERVASIVE.2015.7087075

[228] Daniela Zambarbieri and Elena Carniglia. 2012. Eye movement analysis of reading from computer displays, eReaders and printed books. *Ophthalmic and Physiological Optics* 32, 5 (2012), 390–396. https://doi.org/10.1111/j.1475-1313.2012.00930.x

[229] Peter Ziewer. 2004. Navigational Indices and Full-Text Search by Automated Analyses of Screen Recorded Data Recording Styles : Screen Recording vs . Symbolic Representation. In *Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*. Association for the Advancement of Computing in Education (AACE). https://www.learntechlib.org/noaccess/11112/