Enhancing Interfaces for Scholarly Peer Review

by

Derek M. Cormier

B.Sc., The University of Manitoba, 2013

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

The Faculty of Graduate and Postdoctoral Studies

(Computer Science)

The University of British Columbia

(Vancouver)

August 2016

Abstract

The academic peer review process is the primary means by which papers become a part of established science. For such an important process, the technology we use today to conduct peer review only partially supports the reviewing process and has stagnated behind advances in collaborative document work. In this thesis, we present an overview of the current state of Peer Review Management Software (PRMS) and outline areas where better user support is needed. We then present our broad vision for interactive, web-based manuscript reviewing tools that can be integrated into existing PRMS. Through a controlled study, we demonstrate that providing reviewers with a simple annotation interface is sufficient for identifying 80% of all low-level writing errors in a manuscript before the manuscript makes it through to publication, without proofreading and with minimal additional effort on the part of reviewers. Finally, we present prototypical and conceptual designs for a single-reviewer web interface to address the current lack of manuscript reviewing support by improving the user experience for and efficiency of reviewers.

Preface

This dissertation is an original intellectual product of the author, Derek M. Cormier. The research study described in Chapter 4 was conducted under the supervision of Dr. Kellogg Booth (Department of Computer Science). The study received approval from the UBC Research Ethics Board and is covered by certificate number H14-03428.

The research reported in this thesis was inspired in part by previous research on peer review conducted by Dr. Syavash Nobarany. We have drawn on his overview of the history of peer review for some of the material in Chapter 1, often with additional details provided to provide further context for the new research that we conducted.

Funding for the research was generously provided by NSERC, the Natural Sciences and Engineering Research Council of Canada, under the Discovery Grant Program, and by GRAND, the Graphics, Animation and New Media Network of Centres of Excellence. Facilities and research infrastructure administered by ICICS, the Institute for Computing, Information and Cognitive Systems purchased with funds from the Canada Foundation for Innovation were used for the research.

Table of Contents

List of Tables

List of Figures

Glossary

Annotation – Tacit or explicit markings, notes, or drawings overlaid onto a document for the purpose of reading comprehension, knowledge acquisition for later writing tasks, or collaboration during shared authoring or reviewing

Error noticing – The phenomenon where readers encounter writing errors while reading text without explicitly intending to proofread

Error taxonomy – A categorization of types of writing errors, often accompanied by a distribution of the frequency of each error type

Manuscript – A draft of an academic paper that has not yet been published

Paper – The final, published, and peer-reviewed version of a manuscript

Peer Review Management Software (PRMS) – Highly configurable web-based groupware designed to facilitate the scholarly peer review process for conference and journal publications by collecting author submissions, assigning manuscripts to reviewers, and delivering acceptance recommendations

Report – A report that the reviewer generates after reviewing a manuscript, usually comprising a set of measurements of quality on various dimensions, a plaintext written component, and an acceptance recommendation; typically submitted via a web form

Scholarly Peer Review – The process by which a panel of expert reviewers review author-submitted manuscripts and provide a report and recommendation to the editor regarding whether the manuscript should be published in a conference or journal

Acknowledgements

I thank my supervisor Kellogg Booth for his continual support and mentorship throughout my master's program. His extensive experience and endless supply of anecdotes provided much clarity and context to each problem we tackled.

The MUX (Multimodal User Experience) helped me connect with the broader HCI community and provided an avenue to share and present my work with other researchers. In particular, I would like to thank my colleagues Oliver Schneider, Francisco Escalona, Antoine Ponsard, and Syavash Nobarany for our frequent brainstorming sessions that ultimately helped to shape the direction of my work. I would also like to thank Jessica Dawson and Jung-Wook Kho for thoroughly piloting my user study and providing feedback, which led to problem-free study.

I thank Giuseppe Carenini, who graciously accepted the role of second reader and provided helpful and detailed feedback.

Finally, I am eternally grateful to my parents who always have supported me throughout my education and encouraged me to follow my own path, and to my friends who made my years at UBC all the more fun and rewarding.

Dedication

I would like to dedicate this thesis to my good friend and mentor Chi-Tai Cheng (鄭吉泰), who was unfortunately taken from this world too soon. His love of life and devotion to his students continue to inspire me each and every day.

Chapter 1: The State of Peer Review Technology

Scholarly peer review is the set of evaluation practices and policies used by the academic community to determine which research manuscripts become published, and which do not. Although by its nature peer review is an imperfect process [1–4], it continues to be widely regarded as fundamental for scientific progress and communication [4]. The policies and processes used by peer review to evaluate papers have evolved greatly over the past few centuries. In its earliest days, peer review was left to the sole discretion of journal editors, but with the seemingly exponential advances in scientific knowledge and publications in the twentieth century, we now use panels of expert reviewers from highly specialized fields to review papers, and adhere to policies such as (double) blindedness, openness, disclosing conflicts of interests, and formal rebuttal procedures. Bornmann provides a comprehensive overview of scholarly peer review that summarizes existing research on the process [4].

Peer review practices have evolved over the years; so too has the technology used to carry out peer review. Advances in digital technology have drastically altered how we conduct peer review by enabling a gradual transition away from paper as the primary medium for reviews and correspondence. Instead of sending paper copies of manuscripts, reviews, and acceptance notifications by post, today the stakeholders involved in peer review (authors, reviewers, editors and editorial committees) rely heavily on sophisticated web-based groupware applications to manage online manuscript submission, reviewer assignment, delivery of decisions, and almost every other aspect of the peer review process. The rise over the past two decades of peer review software, advanced authoring tools, and vast digital libraries has gone a long way in making the jobs of each stakeholder more efficient and convenient.

However, while much progress has been made, there are still ways in which current software does not fully support the reviewing process, especially when it comes to reviewing tasks that take place at the level of the manuscript content. Our goal throughout this thesis is to convince the reader that there is much design and innovation left to accomplish before we can consider peer reviewing to be fully supported by computers.

In this chapter, we begin by briefly looking at the history of peer review from the perspective of how it was accomplished using the technology available at the time. We then present an overview of the type of web-based peer review software that is used by academic journals and conferences today. Finally, we outline the areas where software and interface improvements are greatly needed, and discuss the design challenges that remain for future research and implementations to solve. We address some of these challenges directly through prototypes and experimentation, which we describe in later chapters of this thesis. Throughout this thesis, we refer to peer review as strictly for reviewing academic manuscripts, and not in other contexts such as peer review for research grant funding or promotion and tenure consideration. We also make the distinction between a *manuscript*, a research paper that has yet to be accepted, and a *paper*, the final version of a manuscript that has been reviewed and published.

1.1 From Paper to PDF: A Brief History

Today's scholarly review practices date back to the first scientific journal, published in 1665, titled *Philosophical Transactions*. At that time, Henry Oldenburg made editorial decisions about which scientific works to include in the journal to make available for academic discourse among peers in the scholarly community [5]. The first recorded instance where the

opinions of external experts were used to make editorial decisions is commonly believed to have been in 1731, as prefaced in The Royal Society of Edinburgh's journal *Medical Essays and Observations* [6]:

> *Memoirs sent by correspondence are distributed according to the subject matter to those members who are most versed in these matters. The report of their identity is not known to the author. Nothing is printed in this review which is not stamped with the mark of utility.*

Despite the similarity to today's practice of using external reviewers, this type of review policy was not commonplace during the 18th and 19th centuries. Reviews and decisions about publication were often handled internally at the discretion of the journal editor or the editorial board. It was not until the middle of the twentieth century that the standardized refereeing processes that we are familiar with today became the norm, in part due to the growing specialization in scientific fields and increased competition for publication [5,7]. The lack of standard reviewing procedures that included external reviews even in the early twentieth century is exemplified by Albert Einstein's rebuttal to a critical review regarding his submission to the journal *Physical Review* in 1936 [8]:

> *Dear Sir,*
>
> *We (Mr. Rosen and I) had sent you our manuscript for publication and had not authorized you to show it to specialists before it is printed. I see no reason to address the in any case erroneous comments of your anonymous expert. On the basis of this incident I prefer to publish the paper elsewhere.*
>
> *Respectfully,*
>
> *P.S. Mr. Rosen, who has left for the Soviet Union, has authorized me to represent him in this matter.*

Until the end of the 19th century, manuscripts were largely written and illustrated by hand. One of the major difficulties with handwritten drafts was that they often needed to be copied. An author submitting a manuscript to a journal would presumably create a copy for themselves as proof of their intellectual property, or in the case that it was lost during delivery. Although little has been recorded about early peer review procedures and the need for making copies [7], we presume that journals may have required that copies be made in certain situations to distribute internally or externally depending on which (if any) sort of review process was in place. This copying would have been completed either by the author or on the editor's behalf, possibly by employing scribes, and would have taken a considerable amount of time. Transcribing manuscripts, especially those with illustrations, would have been painstakingly slow and error-prone. Typesetting technology such as the printing press or (depending on the period) the linotype, and image copying techniques like block printing or lithography were widespread, but were reserved for publishing and distribution *after* reviewing because they took considerable effort to prepare; they were certainly not used for reproducing drafts.

As typewriters became ubiquitous at the turn of the 19th century, manuscripts and correspondence with journal editors could be compiled far more quickly than by hand, and in a standard and consistently legible document format. Scholars could dictate their work or provide hand-written manuscripts to skilled typists who could produce copies relatively quickly. Copying became even easier with the use of carbonic paper in typewriters, which could produce up to five copies at a time [5], but was necessarily limited to outgoing correspondence. Further technologies such as the mimeograph in 1886 were capable of

producing many more copies, but due to the relatively few number of individuals involved in making editorial decisions few copies were needed, so carbonic paper was likely preferred.

It was not until the invention of the modern photocopier by Xerox in 1949 and its widespread use in academic institutions beginning in the 1960s and 1970s that limitations on copying manuscripts were no longer a limiting factor. The relative ease by which manuscripts could then be written and copied was perhaps in part responsible for the rapid increase in scientific publication within the past fifty years. The increased competition for space in journals led to more comprehensive forms of review involving external reviewers in order to select only papers with the highest scientific merit for publishing. In this sense, the development of typewriters, photocopiers, and other document technology probably helped to bring about the refereeing processes that we use today [9].

In the 1970s, efforts by Brian Kernighan and his team at Bell Laboratories led to the development of phototypesetter software (e.g., the Linotron 202) and typesetting languages such as `troff`—the precursors to modern desktop publishing. The engineers at Bell Labs would use these typesetters to print their academic manuscripts, sometimes to the bewilderment of reviewers who presumed that the papers had already been published [10]. Preparing manuscripts using early, digitized typesetting technology was mostly limited to research institutions like Bell Labs, at least until personal computers with desktop publishing software became commonplace in the 1980s, so much of academia continued to produce their manuscripts using typewriters and to create copies with photocopiers.

Although the origins of what we consider word processing dates back at least to the 1964 IBM Magnetic Tape Selectric Typewriter, which could edit text stored on a magnetic card,

personal computers such as the IBM PC and Apple Macintosh finally brought flexible word processing capabilities to educational institutions and the home user in the 1980s. Word processors enabled authors to edit their manuscripts freely at any location, which was a considerable advantage over typewriters where fixing mistakes or editing could amount to retyping the entire manuscript. The 1980s also saw the development of more functional and accessible typesetting languages such as LaTeX (which remains popular today among scientists and mathematicians), and WYSIWYG word processors such as Apple's MacWrite (later marketed by Claris), Corel's WordPerfect and Microsoft Word that allowed authors to create professionally formatted manuscripts. These software packages provided useful features such as spellchecking and bibliometric tools for managing references.

With the development of many new devices and operating systems, there arose a practical need for a cross-platform document format that could consistently display formatted text on all systems. In 1993, Adobe released the Portable Document Format (PDF) for displaying fixed layout documents, which was built upon their earlier interpreted PostScript language that in turn derived from prior work at Xerox PARC on the Interpress page description language. To this day, PDF remains the *de facto* standard for sharing and distributing academic manuscripts across all platforms.

It appears that changes in the way that scholarly peer review was conducted up until the 1990s were largely governed by the technologies used to communicate written information. The typewriter (and later, the computer) made it easier to write manuscripts, and photocopiers and printers made it easier to produce copies. However, one rather significant inefficiency still remained: the problem of expediently delivering manuscripts around the

world. Despite the availability of the Internet and network protocols in the early 1990s, most manuscripts and reviews were still being sent by post. Authors would typically send three or four copies of their manuscript to the editor for a publication venue via their home institution's mail department. The editor would then mail the copies of the manuscript along with reviewing instructions to external reviewers, who could choose to reject the request and have it mailed back, or to complete a review and have that mailed back. It would take roughly one week to travel from one hop to the next.

The widespread adoption of the World Wide Web in the late 1990s all but removed the inefficiency of delivering manuscripts by post, enabling authors to upload their manuscripts to web servers or send them as email attachments to the editor. The affordances offered by The Web continued to have far-reaching effects. Since the turn of the 21st century, the scientific world has used specialized web software to streamline the peer review process and eliminate many of the administrative pain points for journal and conference editors. In the next section, we summarize the key features and current state of this software.

1.2 Peer Review Management Software

Today, the academic world looks to computer support for managing peer review. We now use web application groupware to manage the entire manuscript review process for authors, reviewers, chairs, and review committees, from enforcing anonymity policies (e.g., double- or single-blind reviews) to coordinating reviews, making editorial decisions, and delivering those decisions to the author(s). These software packages are often highly configurable, allowing for different types of reviews and a variety of access policies to be implemented. We refer to this general category of web software that encompasses both conference and

journal workflows as Peer Review Management Software (PRMS), but in certain contexts these systems have also been referred to in the literature as Journal Management Systems, Conference Management Systems, or Abstract Management Systems.

Some PRMS systems, such as the Open Journal Systems [11], are free and open source, while others such as OpenConf [12] and EasyChair [13] offer freeware versions but require payment for advanced features and hosting services. The majority of these web applications are reminiscent of fairly basic 1990s-era web applications, but some newer proprietary software such as Scholastica [14] and Exordo [15] provide more modern web user experiences. All PRMS tends to support a common set of key features. In the remainder of this section, we provide an overview of these features and how they help solve user problems that existed before the creation of groupware peer review software.

1.2.1 Landing page and branding

Conferences and journals often have a dedicated landing page on their websites containing information (or links to other pages with information) about review policies, submission deadlines, paper format, and event dates. These pages usually have a “call for papers” or registration page that acts as an entry point into the actual PRMS application where authors can register their identities and contact information. While the initial landing page is typically separate from the PRMS, some software, such as the Open Conference Systems [16], has full support for conference and journal homepages, allowing for custom content and consistent styling (headers, footers, and stylesheets) to be used on all pages of the application. This is useful for journals or conferences that want an all-in-one solution or wish to use their own branding or to give their web pages a unique look and feel.

1.2.2 Manuscript submission

After registration, and possibly after an initial abstract submission to initiate the process, the author obtains access to a web form for submitting a manuscript. There are fields to specify each author's contact information, with one author usually being designated as the primary contact (corresponding author) for notifications. The author inputs the abstract (if not done previously), keywords related to the work, and possibly checks a set of boxes to indicate which categories the work falls into; these categories help with assigning reviewers and managing multi-track conferences or journals with multiple areas each with its own associate editor. Some PRMS allow authors to indicate potential conflicts of interest that should be taken into account during the reviewing process.

The author can upload a manuscript file in one of the accepted formats—usually PDF, Word, or LaTeX—and the document is normally required to follow a supplied style template provided on the website (what used to be the work of publishers is now the responsibility of the authors). Some systems accept supplementary files such as original datasets or videos that present the research, the latter being quite common in HCI and computer graphics publication venues. Allowing authors to submit manuscripts electronically removes much of the overhead and inefficiency of sending manuscripts by mail and sorting them into review committees for subsequent distribution to those who need to see them.

1.2.3 Reviewer assignment

Reviewers are assigned to manuscripts either manually or using automatic assignment algorithms. In the latter case, reviews can be assigned based on reviewer bidding—where reviewers can bid on how interested they are in reviewing a particular manuscript—or by

matching manuscripts to reviewers' areas of expertise (reviewers usually indicate their expertise upon signup). In addition to assignment schemes, some PRMS allows certain constraints to be specified, for example, limiting the number of manuscripts per reviewer or adding limits for particular reviewers. Figure 1.1 shows the automatic assignment options that a review committee can select in OpenConf.

Once reviewers have been assigned, they are typically notified by email and then gain access to the submissions they are reviewing through the PRMS, where they can accept or reject the reviewing request. The use of automated assignment makes it far easier for committees to assign manuscripts to reviewers when there is a high volume of submissions. There are often mechanisms to override or augment automatic assignments, which is necessary for unforeseen circumstances such as when there are undetected conflicts of interest or when a

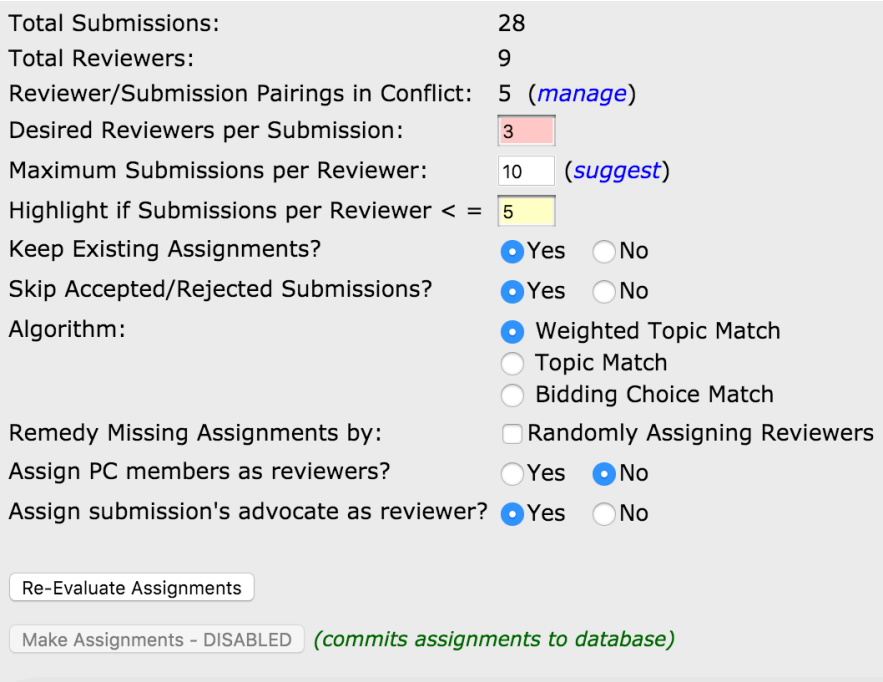


Figure 1.1: Automatic reviewer assignment by a program committee chair in OpenConf. The program committee chair can select a number of options including the desired number of reviews for each manuscript, the maximum number of reviews requested of any reviewer, and the type of matching algorithm that will be used to suggest assignments of reviewers to manuscripts.

reviewer is not able to complete a review or there is not a consensus among the reviewers, so someone else needs to be brought into the reviewing process after it is underway.

1.2.4 Reviewing report

When reviewers accept a request to review a manuscript, they gain access to an electronic review form to create their report (Figure 1.2). The reviewer answers the questions on the form to determine whether they have any conflicts of interest in reviewing the manuscript, to assess the overall quality, novelty, and impact of the paper, and to provide a final acceptance recommendation. The reviewer also typically rates how qualified they feel they are to review the manuscript. The review form contains one or more textboxes that make up the reviewer's

Review

Recommendation:

- Reject: Content inappropriate to the conference or has little merit
- Probable Reject: Basic flaws in content or presentation or very poorly written
- Marginal Tend to Reject: Not as badly flawed; major effort necessary to make acceptable but content well-covered in literature already
- Marginal Tend to Accept: Content has merit, but accuracy, clarity, completeness, and/or writing should and could be improved in time
- Clear Accept: Content, presentation, and writing meet professional norms; improvements may be advisable but acceptable as is
- Must Accept: Candidate for outstanding submission. Suggested improvements still appropriate

Reviewer Familiarity with Subject Matter:

- Low
- Moderate
- High

Relates to the confidence you have in your review

Comments for the Authors:

Constructive comments to the author(s) would be appreciated.

Comments for the Program Committee (authors will not see these comments):

Reasons must be included for all submissions, because they help us determine what to do when reviewers disagree with each other.

Figure 1.2: Example of an EasyConf web review form (some questions omitted for space). A reviewer provides information about the manuscript and the reviewer's opinion and qualifications through a series of radio buttons and free-form text boxes.

written portion of the report in addition to other entries such as numeric ratings or radio button selections from a set of options. The textboxes may be divided into different categories, for example, novelty, writing quality, or general comments for the author, or there may be a single textbox for the entire written portion of the review. A separate textbox is often reserved for private comments meant only for the review committee. The questions that appear in the review form and their widget types (text input, radio button, checkbox, etc.) are usually configurable by the editor or administrator.

1.2.5 Inter-reviewer discussion

PRMS often supports private discussion among reviewers assigned to a manuscript. The discussion mechanisms become available after reviewers have completed their reviews in order to prevent reviewer bias. The discussion provides an opportunity for reviewers to explore potential points of interest or contention. Discussions are typically implemented as simple, linear forum-post-style messages where each entry usually identifies the poster (sometimes only by reviewer number, other times by name) and date of posting. Email notifications are often used for new messages, which the reviewer may be able to respond to by replying to the automated email. An example of a discussion interface can be seen in Figure 1.3.

All Threads »

Discussion

1. Man-Computer Symbiosis
(Paper)

Post Comment - DISABLED

Alan Turing Thu 29 May 2014 02:20:56 PM UTC

Vivamus ac pede lobortis sem tempus convallis. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Curabitur hendrerit tellus non dui. Donec facilisis. Curabitur eu purus sit amet felis suscipit faucibus. In at mi nec est dignissim porttitor. Curabitur tincidunt libero at elit. Morbi tellus. Sed hendrerit, dui a pretium dapibus, neque purus fringilla justo, sit amet luctus quam dui nec dolor. Fusce tortor ipsum, posuere pretium, lacinia ut, viverra nec, erat. Vestibulum nec elit vel nunc consectetuer pretium. Aenean vel lectus. Cras sapien risus, varius in, faucibus et, lacinia sed, enim. Aliquam eleifend turpis id pede. Donec fermentum ultricies leo. Nam convallis metus eget dui. Curabitur a erat. Aenean eu felis sit amet tortor tristique lobortis.

Leonhard Euler Thu 29 May 2014 02:23:39 PM UTC

Donec luctus. Cras gravida. Ut lacinia ligula a velit. Donec elit felis, varius eget, vulputate vel, dapibus ut, arcu. Donec tincidunt massa eu ipsum gravida feugiat. Nam orci nisi, placerat scelerisque, tempor ut, ultrices ac, nunc. Donec ac pede id mauris mattis ultrices. Vestibulum ante nunc, condimentum at, varius vitae, pulvinar vitae, mauris. Duis turpis purus, feugiat nec, pretium at, bibendum a, lectus. Proin suscipit pulvinar est. Fusce id est in dolor dictum commodo. Aenean at sem.

Add a Comment

Post Comment - DISABLED Preview Comment

Figure 1.3: An example of the inter-reviewer discussion interface in OpenConf. Each entry identifies the reviewer who posted the entry and the time and date of posting.

1.2.6 Author rebuttals

Once the reviews and initial acceptance decisions have been made available to the contact author, the author is notified by email and can begin to respond to comments in the reviewer report. The rebuttal is usually entered into a fixed length text box or uploaded as a separate document. Authors are expected to address all of the review reports in aggregate. In a conference, once the rebuttal is submitted the reviewers or the program committee chair makes a final acceptance decision. Some systems (such as OpenConf) support follow-up comments on the rebuttal by reviewers, but these comments are usually only visible to the chair. Systems that support journal refereeing support multiple rounds of comments by both authors and reviewers either within a single submission cycle or through resubmission cycles that continue the refereeing process after authors have revised their manuscripts and resubmitted them, perhaps with rebuttal comments that explicitly address comments from reviewers or that direct reviewers to changes in the manuscript.

1.2.7 Blindness policies

Anonymity of one or more of the participants in the peer review process to others in the process is a common situation. Most PRMS supports either single-blind reviews (authors do not know the identity of reviewers) or double-blind reviews (reviewers also do not know the identity of authors). Whether or not the reviewers can see each other's identity is often a configurable option. Recently, there has much debate and discussion surrounding open review policies, where reviews are not anonymized and are possibly opened up for viewing by a larger community. To our knowledge, mainstream PRMS generally does not provide support for fully open reviewing models. One conference venue supported by the CHI community, alt.chi [17], does support open reviewing but appears to use custom web

software rather than the usual Precision Conference Solutions (PCS) [18] system used by most of the CHI-sponsored conferences. The public accessibility of alt.chi reviews enabled Nobarany et al. to study how non-blinded review policies affect the way reviewers write their reports [19]. They found that there was a need for more interactive communication in order to balance between politeness and clarity in reviews.

1.2.8 Online publication

In addition to handling reviews, PRMS often also supports online publication of accepted manuscripts. Software designed to support conferences allows the creation of a proceedings website that opens up after the conference. It includes a list of the accepted papers, their authors and abstracts, and either a direct link to a PDF or a link to the manuscript's location in a digital library (usually pay-walled). Some software, such as the Open Journal Systems, are designed for the primary purpose of providing open access for the public to accepted papers. Some systems such as PCS can provide a "camera ready" PDF for a full conference proceedings (front matter, table of contents, accepted papers, and indices).

1.2.9 Other support features

Different PRMS packages offer support for a variety of other features that reduce administrative overhead, but are not critical to the peer review process. Such features include support for search engine indexing of the accepted papers, support for assigning papers to conference tracks and scheduling presentations, payment of publication fees through services like PayPal, helpdesk or technical support, and use of automatic plagiarism checking services. OpenConf supports integration with publons [20], an online service for giving credit to reviewers.

1.3 Limitations of Peer Review Software

Peer review software has been effective for organizing peer review for journals and conferences; it reduces administrative overhead by making it easier to assign papers to reviewers, collect reports, handle rebuttals, and deliver final acceptance recommendations. Yet, there are still fundamental reviewing tasks involving all stakeholders that lack adequate computer support and are not addressed by PRMS systems. For example, reviewers are unable to directly link to the text in the manuscript that they refer to in their (plaintext) report, and dialogue between reviewers about parts of a paper (e.g., a discussion about the applicability of statistical methods used) takes place outside of the manuscript context where the issue arises. This can make it difficult for an additional reviewer brought in to easily find parts of the manuscript that relate to the problem and come up to speed in the discussion.

These examples highlight the major limitation of PRMS that we focus on in this thesis: current software does not support interactive reviewing work at the level of the manuscript's content, but instead treats manuscripts only as immutable objects to be copied and passed around to the various stakeholders. The ability to use manuscripts in richer and more interactive ways, both before, during, and after the reviewing process, could help provide better solutions for common reviewing tasks. In this section, we examine a few high-level areas, where PRMS currently lacks support, to illustrate where future interactive reviewing interfaces could be more helpful.

1.3.1 Lack of enforcement for anonymity of authors

In double-blind reviews, authors are instructed to follow a set of guidelines to anonymize their identities. This is one method for promoting fairness in the peer review process

(although its effectiveness has been disputed [1,3]). The anonymity guidelines often include hiding author information, limiting or obfuscating self-references (for example, by referring to the authors' work in the third person rather than the first person), and replacing bibliographic citations of the authors' work with anonymous placeholders. However, it is common for authors to forget about or not fully implement the guidelines. This can result in an unfair advantage or disadvantage for the authors and it compromises the integrity of the process. In order to increase fairness, anonymization of authors should be automated as much as possible and enforced upon submission. For each flagged instance where anonymity may have been compromised, the author should have to manually sign off on each flagged instance and this should be verified by the editorial committee before the manuscript is seen by reviewers. However, this type of in-document flagging and verification is currently not done by any PRMS we know of.

1.3.2 The disconnect between the reviewers' reports and the manuscript

A review often comprises a plaintext report along with a set of quality ratings and the reviewer's opinion or recommendation about the disposition of the manuscript in terms of acceptance or rejection. In contrast to other forms of writing that produce richly formatted documents, the academic community seems to value function over form, and definitely favours conciseness in reviews—written reports are typically no longer than a page, and little advantage is seen in using formatted text. However, the use of plaintext precludes the use of richer functionality such as in-text hyperlinks to connect reviewer's comments with the parts of a manuscript that are relevant to the comments.

Conceptually, a report is intertwined with the manuscript in the sense that the report refers to and summarizes content in the manuscript. As summaries, reports are necessarily generalized, but often there are instances where the reviewer wishes to refer to specific locations in the paper, for example, citing the location of a poorly made argument or an unclear sentence. The author reading the report must perform a manual lookup step in order to find the referenced section of text to inspect or repair it.

The notion of creating anchors to other documents (also known as hypertext) has been around since Vannevar Bush first introduced the concept of a memex [21]—a hypothetical device for organizing collective knowledge—but hyperlinking between text in a manuscript and the review has not been well explored in peer review, likely due to the use of plaintext reports that do not afford bi-directional linking. Instead, "links" often take the form of textual descriptions about where to find the corresponding location in the paper. This is usually done by specifying one or more of the page number, the paragraph number, the line number, a quotation from the paper, or a more general description of the location (e.g., "second paragraph below the Figure 2 in the second column").

The potential benefits of an explicit anchoring mechanism extend not only to pointing out problematic areas, but could also be used by reviewers to clarify or cite evidence for their arguments. It could similarly be used by authors to rebut reviewer arguments with evidence from their paper, which could also free up space in fixed-length rebuttals.

1.3.3 Exclusion of the manuscript as an auxiliary part of the review

In traditional paper-based peer review, the act of reviewing takes place *on* the manuscript. While reading, reviewers use various techniques to structure their knowledge or

understanding of the manuscript that later forms the content of their written report—this can take the form of jotting notes in the margins, tacit markings to remind the reviewer to revisit a section, or many other forms of annotation. This practice persists in the digital world. Some reviewers still mark up paper copies of a manuscript, but more often they mark up the electronic version, at least during the initial stage of their reviewing. While many of these representations of thought and judgment are intended only for private use by the reviewer, there are also certain communicative elements that, if made available, could serve to benefit the author and other reviewers. In a typical conference PRMS, review submissions do not have facilities for uploading annotated versions of the manuscript or for using the annotations in a meaningful way.

Fixing low-level writing errors (typographical, grammatical, etc.) is a good example of how the manuscript could be used as part of the review. If a reviewer sees a large number of writing errors in the manuscript that should be fixed, but the science behind the work reported in the manuscript is good and the reviewer thinks the paper should be accepted, the reviewer could simply leave a remark that the author should proofread the paper more carefully before preparing the final version for publication. However, errors are commonly missed when an author rereads a manuscript and some may make it through to publication. In order to point out the errors to the author, reviewers must write within their plaintext report the location of each error, what the error is, and, if they feel generous, how to fix it, e.g., *"Incorrect verb form on page 6, line 17, change [related] to [relates]."*

This process of writing descriptions of erroneous locations is tedious and time-consuming, which may discourage reviewers from pointing out all of the errors they find, or even doing

anything at all to show the author where the problems are. Worse yet, the detailed low-level error information adds clutter to the main review and may not be necessary for making a decision about acceptance of the manuscript. If instead the reviewers could simply highlight each error they found in the manuscript, and *leave the highlighting in* the manuscript, this could make it immediately clear to the author where the errors were without cluttering the report and without requiring reviewers to do more than they do already, which is to note the occurrence of low-level errors within the manuscript as they read it during their reviewing process. We believe that the activity of pointing out writing errors belongs in the manuscript —not in a separate review document.

Another area that could benefit from using the manuscript as a medium for review is the inter-reviewer discussion that occurs after reviewers submit their initial reports. Usually this is done via a forum-post discussion list on the website as we have previously described. These discussions often revolve around disagreements about the seriousness of various problems with a manuscript, yet are completely disparate from the manuscript itself and thus ignore the surrounding context of the specific problem. Discussions centered in the manuscript could provide context about where an issue occurs and thus could promote better discussions surrounding the issue. Bi-directional links between the manuscript and the threaded discussion could instead be along the lines of Churchill et al.'s anchored conversations [22]. We will further explore this and other possibilities in later chapters.

1.4 Summary

Over the past two centuries, technology has helped move peer review from a hand-written paper-based process to a fully digital process that involves little or no physical documents.

The standard web-based systems we use today excel at managing the overall processing and distributing of reviews, but they lack rich interactions within the document during the various reviewing stages. The report and the manuscript have always been treated as separate entities, but there are ways we can leverage interconnecting both of them to provide more useful reviews and to make the task of reviewing (and responding to reviews) more user-friendly and (more importantly) more useful in terms of the ultimate goal, which is obtaining the best possible evaluations of manuscripts that have been submitted for publication.

In the next chapter, we present our broad vision of how PRMS systems could be integrated with interactive reviewing functionality to make peer review more efficient, fair, and user-friendly, and we outline a set of features that developers and researchers might implement and then fully explore to assess their utility.

Chapter 2: Annotation-based Reviewing Interfaces

Conferences and journals use Peer Review Management Software (PRMS) to manage manuscript submission, distribution of manuscripts to reviewers, and collection of reviewing reports. This software alleviates many of the administrative pain points involved in organizing a publication venue. Until recently, PRMS has focused primarily on supporting these high-level organizational aspects of peer review, and as such, there can be a lack of support for the low-level tasks carried out by an individual reviewer. A reviewer conducting a review of a manuscript and writing a summary report, the most fundamental activity that drives all of peer review, is quite often under supported. This in part due to a lack of dedicated and specialized reviewing tools and the inability of PRMS to interact directly with the manuscript file at the content level, where most of the low-level reviewing tasks take place.

Rather than treating manuscripts as immutable objects passed around to various actors, which is what most PRMS does, in this chapter we explore the idea of overlaying reviewing activities on the manuscript content itself. Our ideas involve extensive use of annotation, so we begin by briefly discussing existing work on annotations and how they have been used in collaborative document work settings. We then discuss how manuscripts might be used more interactively during the review process by outlining a set of preliminary features for future peer review tools. We conclude with a discussion of the need for (and current lack of) specialized reviewing tools to support these proposed features, and how such tools could work in concert with PRMS to provide a more sophisticated environment for interactive peer reviewing.

2.1 Annotations for Collaborative Document Work

"I contend, quite bluntly, that marking up a book is not an act of mutilation but of love."

– Alder Mortimer, *How to Mark a Book* [23]

In the Middle Ages, scholars recorded their musings and findings in the margins of scholarly texts. These marginalia were considered highly valuable to subsequent scholars that they were often transcribed into newer copies of the text [24]. To this day, people annotate documents for a variety of reasons, including expressing new ideas, opinions, and critical remarks, and posing questions about the text, clarifying meaning, and sharing comments with others [25]. The practice of annotating persists in the digital world: we now have vast digital libraries with annotation capabilities and authoring tools that mark up documents to be shared between collaborators. In this section, we give a brief overview of existing research on digital annotations for different use cases.

2.1.1 Annotation as an aid to reading

People often use annotations as an aid to comprehension when reading. Marshall performed an ethnographic study on a large collection of used student textbooks to analyse the form and function of student annotations [26]. The observed annotations took the form of highlights, markings, underlining, notes, and doodling, and were used in anticipation of future attention, place marking, aids to memory, problem solving, interpretive activity, and as a trace of the reader's attention. Other research by Neuwirth et al., Schilit et al., Wolfe, and O'Hara and Sellen have shown that annotating while reading is beneficial to structuring and recalling knowledge for later writing activities, it decreases the tendency to unnecessarily summarize, it improves understanding, and it aids visual search and active reading [27–30]. Much of the

research alludes to the suggestion that digital libraries ought to support annotation tools to encourage active reading and the sharing of ideas [26,29,31], with Kopak and Chiang actually implementing a set of such features for use when reading papers in the Open Journal Systems [31].

2.1.2 Annotations for collaborative work

Annotations have great value not only as a personal tool, but also as a medium for supporting collaborative authoring and reviewing. Neuwirth et al. found that there was a need for in-document support for social interaction among collaborators, the workflow of drafting ideas into writing, and the inclusion of remote users. To address these needs, they introduced the PREP editor for simultaneously planning, commenting, and displaying content [32]. To address the need for effective communication among collaborators, many prototypes have been developed that support in-document communication via threaded conversations in the form of annotations, usually anchored in some way to the text itself [22,32–35].

In their study of a large corporate workgroup's use of web annotations for document collaboration, Cadiz et al. found that there was a need for prioritization and better notification systems for large groups—people wanted more context in their email notifications in order to understand what had changed in a document, and whether the change was relevant to them [33]. Further, participants tended to focus less on low-level details such as grammar because that type of annotation might be distracting to the larger group. Certainly, in larger systems that support larger groups it does seem necessary to have the ability to hide less important annotations. Glover et al. have written a comprehensive overview of other web-based systems that support annotation [36].

Other work has taken collaboration features further, introducing explicit annotation roles and access control, progress tracking, version control of annotations, one or more status settings for individual annotations (resolved/unresolved, read/unread, etc.), annotation grouping, increased group awareness, and support for meta-commentary and decision making within the document context (as opposed to, for example, having discussion about issues through email) [34,35,37]. There is also a relevant body of work on e-learning tools that uses annotations to support collaborative learning, for example, in course materials or online tutorials. In e-learning applications, annotation research focuses on ways to tag and organize comments, along with novel ways of displaying the annotations alongside content.

2.1.3 Annotation taxonomies

With the many forms and functions of annotations that exist in different contexts, researchers have attempted to formalize ways to describe them [25,35,37–41]. In Table 2.1 we present a condensed and generalized overview of the various properties of annotations that others have described or implemented.

Table 2.1: An overview of the various properties of annotations described in the literature.

Property	***Description***
privacy	private annotations are meant only for the creator's use, whereas public annotations are meant to be shared with other people
scope	the extent of the annotation's audience, e.g., global, institutional, work group, or personal
formality	informal annotations refer to ones that involve tangibly manipulating a document, e.g., highlighting or marginal comments, whereas formal annotations refer to those that describe relations or grouping among other annotations or contain meta-data about annotations
tacitness	tacit annotations are highly personal or incomplete representations of work such as markings or drawings that may be unintelligible to others; on the other hand, explicit annotations have a clear and readily-understood interpretation
for writing vs. for reading	some annotations intended for writing tasks, e.g., marking a sentence to quote it later, while other annotations are meant for reading comprehension, such as colourful highlighting or underlining in textbooks
transience	the lifespan of an annotation; e.g., comments made by a co-author reviewing a draft are later resolved
context	the part of a document the annotation belongs to, e.g., a span of contiguous text, a numbered section, a geometric region, an image, or the document itself
metadata	data or attributes that belong to an annotation, e.g., creator, recipient, timestamp, category, comment, rating, priority/urgency, resolved state, read status

2.2 A List of Features for Future Peer Review Systems

There has been little exploration into how manuscripts could be used more interactively as a medium for bi-directional information exchange throughout the reviewing process. This comes as no surprise—manuscripts are usually submitted as PDFs, and writing software that parses or manipulates PDFs to add information in ways that were not originally intended is difficult. Placing implementation considerations aside for the moment (with further discussion in Section 2.3), software that can work with manuscripts on a deeper level has the potential to improve fairness, efficiency, and the overall user experience of reviewing. We have identified three broad areas where augmented manuscripts could be used to help facilitate the reviewing process: automatic processing of manuscript content, using annotations for interactive private report writing, and shifting collaborative reviewing

activities onto the document context. With these in mind, we now outline a preliminary list of features that future peer review systems or external reviewing tools should aim to support.

2.2.1 Automated or semi-automated anonymization

Where possible, submitted documents should be automatically anonymized upon submission. Author information and bibliographic citations might either be removed or replaced with anonymous placeholder text or blanked out with a solid overlay. Where possible, special considerations should be taken to preserve the manuscript's original spatial layout. Any hidden metadata stored within the file, such as the name of the owner or their computer, annotations, etc. should be automatically cleared. For more subtle compromises of anonymity that are not easily searchable, such as self-references to an institution or previous work, intelligent algorithms should be developed to detect possible issues and flag them, for example, by looking for text where personal pronouns are in close proximity to citations. The author and chair should be notified of these potential problem areas to ensure they are fixed before the manuscript is seen by reviewers, possibly by ensuring that the authors manually "sign off" on each detected instance. Only the final anonymized version of the manuscript should be sent to reviewers.

Current Level of Support: Currently, there is no support for automatically anonymizing document content. Microsoft Word has a Document Inspector feature that can strip a document of metadata including personal information, annotations, and hidden content. In a simple proof of concept, we implemented partial anonymization features for LaTeX documents via scripts that intercepts the LaTeX chain of commands and generates anonymous placeholders for bibliographic entries and author information (see Appendix A).

2.2.2 Hyperlinking from the report to the manuscript

Reviewers should able to include hypertext links in their reports that refer to specific locations in the manuscript in order to guide readers of the report to that location. These in-context links could be used by reviewers to provide backup for their assertions or to point the author to areas that need to be improved. Similarly, authors could use links to help support their rebuttal points. Currently, reviewers or authors must resort to tediously written textual representations of document location such as *"see page 4, paragraph 7."* Readers should be able to click on a hyperlink in the report and be immediately brought to the corresponding manuscript location. Hyperlinks should be capable of linking to different entities on the manuscript such as words, sentences, paragraphs, heading, pages, or figures, and it should be clear upon opening the manuscript to which entity the hyperlink it anchored to, perhaps using visual effects such as animated glows or highlighting.

Current Level of Support: SyncTeX [42] is an add-on to the LaTeX compilation chain that allows bi-directional linking between locations in a .tex source file and the corresponding location in the output PDF, but requires support by the PDF reader and text editor to actually perform the navigation; this support is not offered by mainstream software like Adobe Reader. Bi-direction linking using SyncTeX is supported in ShareLaTeX [43], a web-based LaTeX authoring environment for synchronous editing by multiple authors, and by PDFTron's Xodo [44], a web-based PDF viewer and annotator, that supports links to any annotation in their Slack integration tool [45] (which is currently in beta testing).

2.2.3 Specialized annotations for reviewing

A future reviewing tool should support common annotations that people are familiar with, such as highlighting or anchored notes, but should also support new annotations that are specialized towards reviewing. For example, a "missing reference" annotation could provide a direct link to a paper that the reviewer thinks should be included, and could be conveniently inserted via a lookup from the reviewer's own BibTeX file. A "checklist" annotation could be used by reviewers to create a to-do list containing points the author must address in the next draft (possibly with each point anchoring to a spot or other annotation in the manuscript); the author would need to check off each point as an acknowledgement that the point has been noted and addressed.

Reviewers should have the option of classifying annotations by the issue(s) they represent, for example, a novelty, methodology, or writing issue. This classification could be useful for providing insight into the frequency of various types of issues, which the reviewer could use in the summary of their report. For example, if a reviewer notices that 75% of the annotations they created were classified as methodological issues, they may want to spend a proportional amount of their report discussing those issues.

For recurring issues in a manuscript, it may be useful to give annotations a formal structure that relates them to one another (similar to Zheng et al.'s Structured Annotations [35]), or allow single annotations to be anchored to multiple non-contiguous locations within the manuscript to avoid repetition for reviewers and to provide a more compact list of comments to authors. A reviewer should be able to choose which annotations are meant for private

reviewing activities (to be purged upon submitting the review), and those that are meant to be shared either with the author, other reviewers, or the chair.

Current Level of Support: Most mainstream authoring software and PDF readers/manipulators support only general annotations on documents such as highlights, comments anchored on contiguous sections of text, or drawing on the document.

2.2.4 Auto-generating textual descriptions of writing errors

Some reviewers wish to help the author improve the writing quality of a manuscript, especially if they intend to provide an acceptance recommendation and the work is otherwise high quality except for some aspects of the writing. Often the only way the reviewer can currently communicate instances of errors to the author is to write out textual descriptions in the report, for example:

> *Spelling error on page 6, just below the figure; change "concent" -> "consent."*

The reviewer should be able to capture the error within an annotation (such as a highlight), possibly indicating the type of writing issue, and the description of where the error occurs should be auto-generated and appended at the bottom of the report. This feature assumes that the review process in place only supports textual reports and does not permit an annotated manuscript to be sent back to the author, which would be preferable, but short of that there should be support for auto-generating most of the textual descriptions of suggested wording changes.

Current Level of Support: There is currently no support for auto-generating the descriptions of writing errors. Reviewers must type out the descriptions manually.

2.2.5 Report outlining using annotation visualizations

Acquiring knowledge about a source document and structuring that knowledge is key for the writing process [27]. People often create annotations (comments, tacit marks, etc.) as an aid for understanding and expressing their thoughts on the material [28]. This extends to peer reviewing where a reviewer may mark up a manuscript with annotations to record their thoughts and judgments that will later be expressed in the summarized report. However, the way that annotations are scattered around a manuscript is often not conducive to organizing the issues they represent into an outline. Neuwirth et al. found that it was important for writers to be able to visualize their notes, perhaps in a graphical network, to facilitate writing [27].

Reviewing tools should support reviewers in outlining their written report by allowing them to further organize their annotations (which are anchored within the manuscript) *outside* of the manuscript context. They should be able to sort and filter annotations by different attributes such as the type of annotation or category of issue they represent, manually group related annotations together and move them into ad-hoc or hierarchical structures or outlines, and quickly jump between these structured views and a particular annotation's context within the document. Annotations containing written content (such as comment annotations) should be directly insertable into written report (possibly with some massaging afterward to fit within an existing paragraph). The written report should be augmented into these structured annotation views so that the reviewer's organization of their annotations can be viewed in parallel with their report drafting.

Current Level of Support: Previous research has looked into how people can anchor their thoughts into a set of searchable and classifiable notes for later lookup [27], or provide visual overviews of the annotations in a document [46]. However, no work to our knowledge has explored ways these annotations could be manipulated directly into an outline. No mainstream editing tools support the transfer of in-document annotations into an outline structure to facilitate report writing.

2.2.6 Sharing annotations with the author

While some annotations are meant for private report-writing activities, others are meant to be shared with the author and do not necessarily need to be summarized in the written report nor persist in a shared version of the manuscript that other reviewers can see. Such annotations may include highlighted instances of writing errors, or notes containing suggestions on how to better organize or frame the work. The reviewer should be able to designate that certain annotations are intended for sharing with the author and are thus to be preserved within the manuscript as part of the delivery of their review to the author, whereas other annotations could be designated as being private to the original reviewer by default.

Current Level of Support: Most conference and journal PRMS does not allow the reviewer to upload marked up manuscripts. This prevents reviewers from sharing annotations with the author. In some journal processes, once a paper has been accepted there can be a back-and-forth between the editor (or other reviewers or editorial staff) and author where they may send marked up manuscripts between each other, but this is not the norm during the primary reviewing process.

2.2.7 In-document inter-reviewer discussion

Reviewers should be able to engage with one another in threaded discussions within the document context, similar to Churchill et al.'s anchored conversations [22]. The discussions should be able to be anchored to different parts of the manuscript, including sentences, paragraphs, headings, figures, or some arbitrary coordinates on a page. This could take place on a centralized source document (similar to Google Docs [47]) or in an asynchronous manner for offline reviewing. In-document commenting should be augmented with the out-of-document commenting that already occurs in PRMS. For example, discussions occurring in the document could still be represented in a list of discussions topics on a page within the PRMS, but clicking on it might open up the manuscript to the discussion annotation's anchor point. For conferences or journals supporting more open reviewing policies, these discussions could possibly be made public after the reviewing process has completed.

Current Level of Support: The idea of in-document discussions is not new [22,33–35,37,41], but it has not yet made its way into academic peer review. Some peer review systems support reviewer discussion through forum-post style discussions built into the web application, separate from the document (see Section 1.2.5); others support discussion via anonymous emails. Word supports discussions through single-threaded comment annotations, and LaTeX has a series of packages (pdfcomment [48], todonotes [49], and TrackChanges [50]) that allow creation of graphical, marginal, or in-line comments that appear in the outputted PDF. For both Word and LaTeX, these comments are shared asynchronously, usually by emailing the output file. Google Docs, Xodo, and PleaseReview [51] support synchronous comment making on a source document, but are limited to single-threaded comments.

2.2.8 Filtering annotations

With a large set of annotations for a document, there should be ways to filter them, depending on who is working on the manuscript and the context. Reviewers drafting their report may want to filter their own annotations based on the type of issue (e.g., novelty of the science, citations to the literature, grammar, etc.) for the current paragraph they are writing, or filter them by private versus public status to double-check what the author will be able to see if annotations are shared along with the report. Reviewers engaging in discussion on a manuscript may want to apply a filter that shows only new comments they have not read or follow-ups by other reviewers on earlier comments they themselves made. Authors may want to filter annotations returned to them to separate writing errors from methodological issues when revising their manuscript. Reviewing tools should provide support for these types of context-based filters.

Current Level of Support: Word, Google Docs, and Adobe Acrobat Reader include annotation meta-data indicating who created an annotation and what time is was created. Word is able to filter annotations by their creator or by type (comments, insertions, deletions, etc.), and Xodo can filter by the time created or the text written in comments (their beta Slack integration tool can further filter by recent activity and new messages). However, to our knowledge no mainstream tools support the classification of annotations and filtering by semantic types, e.g., methodology issues or novelty, which is especially relevant for peer review.

2.3 The Need for Dedicated Reviewing Tools

Advances in collaborative authoring have led to feature-rich word processors such as Microsoft Word that have annotation capabilities, and web-based synchronous authoring tools such as Google Docs and ShareLaTeX that allow collaborators to simultaneously work on a centralized source document. Although authoring technology has continued to advance so that tools such as these are used extensively by authors to collaborate on manuscripts prior to submission, the tools are not especially useful to reviewers who receive only PDF versions of manuscripts. Furthermore, the annotation capabilities of traditional PDF readers and manipulators such as Adobe Reader, PDF Annotator [52] or Foxit Reader [53] only provide general annotation capabilities that are not specialized for academic peer review and, worse yet, different tools may use incompatible data schemes for storing annotations within the file. Recently some software has been developed to support annotating and engaging in discussion on PDFs over the web [44,51], but there is currently no support for integrating annotations with report writing.

Some dedicated reviewing tools have been developed for very specialized domains. For example, Mechanical TA was developed for and is used extensively by UBC's Computers and Society (CPSC 430) course to support teaching students how to become effective reviewers by reviewing and grading their peers' essays [54]. However, to our knowledge there is no dedicated software for academic peer review of manuscripts that supports the full set of features and interactions discussed in this chapter. Moving forward, we need to design specialized reviewing tools for academic peer review to support rich annotations that may be shared with the author or other reviewers, and that can integrate with existing PRMS systems.

2.3.1 Choosing PDF as the standard reviewing medium

From a practical standpoint, it makes sense to choose PDFs as a standard medium for interactive reviewing because it would be difficult to develop and maintain consistent annotation and reviewing features across different document preparation systems, especially when many are proprietary and their plugin functionality may be limited. PDF is the standard document presentation format that all mainstream document-authoring tools are capable of exporting to, and as such, future reviewing tools should at the very least support PDF input.

2.3.2 The case for a web-based review tool

We argue that the web is the most suitable platform for developing an interactive peer reviewing tool because (a) open source JavaScript projects enable rendering PDFs in an HTML representation, allowing us to implement virtually any type of annotation supported through DOM manipulation, and (b) a web-based solution could be seamlessly integrated into PRMS, which are also web-based.

Initiatives for supporting viewing and manipulating PDFs in a browser have led to projects such as Mozilla's open source PDF.js [55] and PDFTron's proprietary WebViewer [56] for rendering PDF documents in HTML5. While WebViewer aims to support all PDF features and uses a full-stack infrastructure to generate renderings for HTML5 canvases, PDF.js is a frontend framework that uses JavaScript to render onto canvases, but does not yet support all PDF features such as forms or certain methods of rendering. However, unlike WebViewer, PDF.js uses transparent text overlays as DOM elements on top of the rendered canvas to enable the user to natively select text in the browser. This access to selectable DOM elements will be crucial for developing web-based reviewing tools because we can use these elements'

text nodes as anchors for annotations using methods similar to those described in Appendix B. For now, access to text nodes is a good starting point. We hope that the PDF.js development team and community contributors will continue to add more native HTML structural representation of PDF documents, such as images or figures, in non-canvas HTML tags.

Developing a web application has a few key advantages over a desktop solution. First, we can avoid dealing with the internal implementation details of PDFs, such as finding a way to represent our custom annotation data within a PDF and creating a custom application to render and create these annotations. Instead, we can store annotations on a web server based on the necessary information needed to reconstruct the annotations into a rendered web PDF (without necessarily needing to store the text content of the anchor [25]). Second, browsers are already designed to support rich interfaces and interactions so all the tools needed to create functional and visually appealing annotations are mostly already present. Finally, a web-based solution that operates on DOM elements could be extended and specialized for other collaborative document work applications that use a browser as a platform.

Embedding a web-based reviewing tool within a PRMS opens up further possibilities. The tool could be loaded with specific subsets of functionality depending on the stage of review and who is using it, and only with the annotations that are accessible to the user. For example, a reviewer conducting an initial review could open the manuscript in “review mode” which has the subset of annotations used for private reviewing activities, and these annotations would be the only ones accessible. After initial reviews, reviewers could open up the manuscript in a “discussion mode” that only supports basic anchored conversations and makes the annotations visible to but not alterable by the other reviewers. Hyperlinks within

pages in the PRMS could act as entry-points into the tool. For example, reviewers could possibly see a list of discussion topics occurring within the manuscript and clicking on it could open the reviewing tool at the location of the discussion. Authors reading their reviewer reports could click on hyperlinks left by the reviewers and be directed into the reviewing tool, which opens the manuscript at the sentence, figure, or other object specified by the hyperlink. Perhaps the author could open the tool to see annotations left by all reviewers (with options for filtering), supplied by the backend of the PRMS being used.

2.3.3 Necessary extensions to PRMS

A number of extensions will be needed to PRMS to embed the reviewing tool, which could still be developed as its own standalone component. The PRMS will need access control mechanisms for annotations created on a manuscript. For example, public annotations created by a reviewer should be visible to the author, but not the private annotations that were created while reviewing and drafting the report. Similarly, group discussion annotations created through inter-reviewer discussion may also need to be hidden from the author depending on the openness of the review process.

Extensions will be needed to support automated anonymization and cycling through flagged instances of problems. When an author uploads a manuscript and the content is scanned for possible anonymity violations, the PRMS system will need to manage this extra state to keep track of whether a paper has been uploaded but not signed off as being ready for reviewing. The anonymity reviewing tool could be used by both the author and chair to double-check flagged areas and the PRMS would not send the manuscript to reviewers until this check has been completed.

It may be useful for PRMS to collect data about the frequency and types of annotations created by reviewers using the review tool, for insight into papers that have significant problems, or as data for studying the nature of errors or error-finding while reviewing academic manuscripts. In the former case, if many writing errors are flagged by the different reviewers, then the chair may want to bring in an additional reviewer or proofreader to help the author with writing quality. In terms of research, collecting data about the types of annotations that reviewers create could provide insight about how to improve the tool's design. This data collection should be done only with the express permission of authors and reviewers.

2.3.4 Integration with authoring tools and a potential use case for ShareLaTeX

So far, we have worked under the assumption that the authoring process is completed before submitting the manuscript to a PRMS. However, there is a case to be made that integrating authoring tools earlier into the process could yield certain advantages. One such advantage is access to the author's source manuscript document. Source documents, such as LaTeX or Microsoft Word files, contain structure that may not be preserved in the final PDF. For example, a .tex file contains an author section and can link to a BibTeX file for populating the bibliography. Easier access to this structural information could make tasks such as automated anonymization easier and more precise. Rather than attempting to remove identifying information from a PDF, certain information could be automatically removed from the source upon submission and that version of the source could then be compiled into a PDF for reviewing. The IEEE VGTC LaTeX template can already do this to some extent by allowing the author to turn on a "review" flag that disables the copyright and suppresses the authors, affiliation, and acknowledgements [57].

Integration with authoring tools may also be helpful for reviewers who want to make changes directly to the manuscript while reviewing rather than creating annotations on an immutable version. This would be more akin to collaborative co-authoring where a reviewer creates tracked changes for the authors to accept or reject. This may be most useful for journal submissions where refereeing is more of a back-and-forth exchange between the authors and the referee than it is for conference reviewing. Collaborating on the source document in this way would enable better change tracking and history than taking changes suggested in a PDF manuscript and transferring them back to the source document. This possibility of integrating web-based authoring and reviewing tools raises many complex implementation considerations and questions that are beyond the scope of this thesis.

One environment that could be used for experimenting with integrating authoring tools into PRMS is ShareLaTeX, a popular synchronous co-authoring web application that enables authors to simultaneously work together on a LaTeX document. Similar to our proposed interactive reviewing tool, ShareLaTeX uses PDF.js to output a browser-rendered PDF so existing reviewing functionality could still take place there. ShareLaTeX has access to the authors' BibTeX file, so it could be used to produce automatically produce PDF manuscripts with anonymized bibliographic entries. We explored some preliminary solutions for automatically anonymizing PDF manuscripts with access to a source LaTeX document and BibTeX file, which are documented in Appendix A.

2.4 Summary

Reviewing is an activity that conceptually takes place on the manuscript, yet reviewing work, including report writing, is done using disparate and oftentimes unsuitable tools designed for

authoring or PDF manipulation. The next step for peer review technology is to design interactive and annotation-capable reviewing tools that overlay directly onto or integrate with manuscripts and that support reviewing for later report writing. In this chapter, we outlined a list of features that designers and developers of future peer review systems should aim to support or expand upon. These features point to the need for a web-based reviewing application that uses DOM manipulation of manuscripts to support rich annotation functionality. There may also be advantages to integrating collaborative authoring software, such as ShareLaTeX, into the reviewing process.

Chapter 3: Low-level Writing Errors

The academic peer review process is often referred to as the gatekeeper of science [1,2,4]. Its primary goal is to ensure that only high-quality research is included in the compendium of scholarly knowledge and disseminated across the academic community. Not only should the science described in a paper be good, so should the writing in the paper—after all, the purpose of a paper is to communicate and share knowledge, and a well-crafted article aids the reader's understanding (and enjoyment) of the written work. However, clear explication is not always a surety in academic writing. As language expert and experienced editor Steven Pinker noted, "Too few academics have the ideal of clear, classic prose, where the writer and reader are equals and the reader can see what the writer is seeing" [58].

One particular problem in academic writing is that low-level writing errors such as spelling, grammar, and formatting errors have a knack for making their way into academic papers and, much to the dismay of authors, they are often only discovered post-publication. In this chapter, we discuss what the literature tells us about the types and prevalence of low-level errors in academic writing, why the peer review process is not robust at catching them, and some of the existing barriers to addressing this problem. We then propose a practical method for catching writing errors early on, during the review process, by asking reviewers to catch low-level errors while minimizing any added work on their part. This is accomplished using a novel lightweight interface that we describe at the end of this chapter. In the next chapter, we present a study of the effectiveness of the interface in which we measure error detection performance.

3.1 Writing Errors in Scholarly Publications

Anyone who has been involved in scientific writing, whether as an author, a reviewer, or a reader, has no doubt experienced the problem of low-level writing errors showing up in every stage of the process. In this section, we identify the specific issues that we will later try to address in a tool to support reviewers and authors in the task of identifying low-level errors.

3.1.1 What we know from the literature

There is a small body of work that looks at writing errors seen in scholarly works. Pollock and Zamora used their automated SPEEDCOP algorithm to search for spelling errors in a large collection of text from seven scholarly databases [59]. The algorithm detected incidences of incorrectly spelled words by using dictionary lookups, so it could only account for incorrectly spelled words but not correctly spelled words that were used incorrectly (often referred to as malapropisms). They found that 0.2% of all text contained spelling errors, and that this percentage was consistent across the databases. To put these findings in perspective, a typical eight-page paper with 43,000 characters could be expected to have as many as 86 misspellings based on Pollock et al.'s estimates. It is important to note that Pollock et al.'s study (and the materials used) came before the widespread use of spell-checking software, and thus the findings may not be representative of the frequency of misspellings in today's scientific papers, although given our experience reviewing papers we suspect that things have only marginally improved since their study.

Another highly problematic and prevalent source of writing errors in academia is bibliographic entries. Lists of references at the end of a paper often include misspelled author and journal names and incorrect dates, pages or volumes. Sweetland [60] found that in the

worst cases, authors have had their names misspelled (or left out), with the misattribution taking years to resolve. Sweetland provides an extensive literature review on the topic of incorrect bibliographic entries and reports that in studies of the medical literature, between 29% and 54% of citations have been found to be erroneous [60]. Unlike spelling errors, which tend to occur independently, it is very easy for bibliographic errors to be propagated from source to source.

Beyond spelling and bibliographic errors, there is to our knowledge no prior work that investigates the prevalence of *other* types of errors found exclusively in academic writing. This comes as little surprise, because understanding the types of errors authors tend to make requires large-scale studies of written materials—a time-consuming task that, at least until recently, lacks obvious automated solutions. As a result, we are left with an incomplete picture of how often grammar, formatting, or other types of errors actually occur in scholarly publications.

3.1.2 Why writing errors make it past peer review

One possible reason that writing errors tend to not be fixed during peer review could be that the way papers are written has changed during the half-century or so that computer science has been a recognized discipline. Conferences (which, in some areas within computer science, sometimes rival journals in prestige and reputation) are fast-paced and have strict and time-constrained deadlines for submission, review, and final preparation of manuscripts. This almost certainly leads to a large number of quickly written (and thus error-prone) papers submitted just prior to the deadline. This is not to say that error-free writing is not encouraged—in fact, many publication venues explicitly require careful proofreading by

authors and they even provide proofreading guidelines and tips [61]. However, unlike journals, conferences usually lack a dedicated copyeditor for the final version of a paper (this is due to economic reasons), and thus they do not conduct final quality checks on the writing except on an ad hoc basis. Furthermore, once a paper has been accepted there may be little incentive for authors to expend great effort to find and fix potential writing errors because the written paper is perhaps seen as just an adjunct to the oral presentation that will be given at the conference. Authors are under time pressure to submit the final version of their papers and they are aware that little or no further review of the paper will take place. In some cases, English as a Second Language (ESL) writers may not be capable of easily writing error-free papers on their own. More importantly, some authors may also hold the false assumption that any lingering errors in the manuscripts that they submitted will already have been found and pointed out by reviewers by the time they receive acceptance notices.

Relying only on reviewers to catch low-level writing errors is at best questionable, especially for conferences. The process for conferences is not only fast-paced and constraining for authors, but also for reviewers, who usually provide one round of reviews and then perhaps see rebuttals after which final decisions are made on acceptance or rejection with no middle ground for paper that are "almost" ready for publication. Reviewers for journals and for conferences are selected based on their technical expertise in their fields, and, as such, tend to focus their efforts on issues of research methodology, novelty of the scientific findings, and potential impact of the work under review. With copyeditors removed from the equation, the only obvious replacements—the reviewers—often do not have time to perform extensive proofreading, and are in fact usually not expected to help with improving writing quality beyond global assessment of an acceptable level of writing quality. While a paper with an

egregious number of errors is likely to be rejected, we suspect that strong papers with a moderate or lower number of errors are likely to make it through the reviewing process, and that due to the expectations and time constraints of reviewers, not all of these errors are caught until after publication, if then.

While time constraints are a concern in conferences and may exacerbate the frequency of writing errors, journals usually spread the reviewing process over longer periods of time and over multiple reviewing cycles. Having more rounds of reviews and editing should mean that errors are more likely to be caught, yet even journal papers suffer from post-publication blemishes. The crux of the issue is that there are fundamental limitations on human proofreading capabilities, and there will always be some proportion of errors that even a tenacious author will not find.

There is some tangentially related literature that looks into the performance of proofreaders, but only to a limited extent and without being the central focus of the studies.

In their work looking at fatigue effects between reading on paper and on a CRT monitor, Gould and Grischkowsky discovered that proofreaders could find between 67% and 70% of typographical errors, respectively, with no significant differences between the paper-based and monitor-based groups [62]. Rouet reports on multiple studies that compared errors caught by annotating on paper versus using a stylus on an electronic document, and found that there were no significant differences in proofreading ability between the stylus and pencil groups nor the types of errors found [63]. In their Crowdproof tool, a Microsoft Word plugin for crowdsourcing work to be done on a document, Bernstein et al. found that they were able to catch 67% of spelling and grammar errors using the combined input from a large

number of Mechanical Turkers. When this was paired with Word's built-in spell-checker, 82% of the errors across five source documents were caught [64]. These results tell us that for an average single proofreader, we might expect the proofreader to miss at least 30% of the errors in a manuscript, and that even a group of proofreaders combined can still miss a significant proportion of errors.

3.1.3 Barriers to asking reviewers to proofread

Despite limitations in proofreading ability, having more passes and more proofreaders ought to significantly reduce the number of writing errors that persist into the final version of a manuscript. One solution to reducing errors in academic papers is to ask reviewers to proofread papers alongside their regular reviewing duties. However, whether writing quality should be incorporated as a part of peer review is a divisive issue. Not everyone agrees that reviewers should be responsible for finding writing errors—Nobarany et al. report that a lack of time in general on the part of reviewers and a dislike for being asked to review a paper with writing issues are major demotivators for accepting review requests [65]. This is primarily because reviewers are busy people—often they are professors, graduate students, or professional researchers who agree to review papers on their own time and typically without monetary compensation [66]. Proofreading in addition to reviewing can take significantly more time than simply reviewing a paper, which accounts for the demotivating effect.

There are also significant technological barriers that discourage reviewers from catching errors while performing a review. Web submission systems such as OpenConf and EasyChair require that reviews be entered into web forms with radio buttons (for quality-of-reviewer expertise measures) and text boxes (for the written report). Pointing out low-level writing

errors using these interfaces is time consuming. If reviewers want to indicate to the author that they have found an error, they must write a textual description including (a) what the error is, (b) where to find the error (e.g., page and line number), and optionally (c) how to fix the error. For example, a textual representation of a report of a typographical error might look like this:

```
Typo on page 5, line 25; [recieved] -> [received]
```

Pointing out errors using textual descriptions is highly tedious, and a high number of errors in a document could make many reviewers unwilling to put in the added effort.

3.2 A Practical Solution for Catching Errors during Review

Taking into account the issues just discussed, we can suggest a compromise that may at least partially address the problem of low-level errors that can be implemented without excessively burdening reviewers with additional work.

3.2.1 Asking reviewers to catch errors

As we have seen, writing errors do make it into published papers. This phenomenon is not limited to ESL or inexperienced writers—even highly skilled authors are prone to making mistakes. So absent professional copyeditors, the peer review process is really the last line of defence for finding these errors before publication. Although we can continue to recommend that authors carefully proofread their work or acquire outside assistance if they feel they are not able to do a good enough job, we cannot rely on them to find all mistakes. If we do not have dedicated copyeditors, then *someone* needs to do the work, but who? Some ESL writers feel anxious about their writing ability and resort to using proofreading services, which can

incur a hefty fee [61] (an online search for proofreading services priced this thesis at $540 USD with a one-week turnaround). Another source of external assistance could come in the form of crowdsourcing the error discovery process [64,67], but this work is in its infancy and it is not clear that it will scale to the level that would be required for peer review. Instead, we look at solutions that can be deployed within the existing peer review system and that are likely to be scalable.

As a first step, it seems reasonable to keep things internal to the existing set of people involved in peer review and ask reviewers to help ensure writing quality. The inclusion of separate roles for reviewers, where one reviewer may focus primarily on writing quality, has been proposed but not fully explored [65]. Any solution involving reviewers must carefully consider the cost of reviewer time. While some reviewers have expressed a desire, or at least a willingness, to help with detecting writing errors [67], it would seem that requiring reviewers to proofread as a general rule is untenable due to time constraints, so if we can discover ways to decrease the time or effort needed to help detect low-level writing errors, perhaps we can provide a middle-ground solution that strikes a sensible balance between the cost of extra reviewer time and the benefits of improvements to a paper. With this motivation, we explored ways to divide the labour in a way that requires minimal work from reviewers, yet still results in effective error detection. We identified one technique that capitalizes on a common side effect of reading—a person's perceptual ability to *notice* errors while reading.

3.2.2 Error noticing: an alternative to proofreading

While reading a paper, the reader occasionally comes across a spelling, grammatical, or formatting error. Unless it is an amusing malapropism or something that seriously impairs

understanding, the reader moves on rather quickly. Unlike proofreading, where the reader enters a cognitive mindset for finding errors, this *error noticing* activity occurs naturally while visually consuming text. While we presume that proofreading is much more thorough in locating errors, it not clear exactly how the two methods of error detection compare in terms of overall performance if the metric is simple error detection rather than full error correction. We can also ask how well a group of readers might perform. For example, could a set of three readers (or reviewers) *notice* more errors while reading than a single dedicated proofreader would while carefully scouring the text?

The notion of catching errors simply by noticing them is attractive for peer review because it implies that only a little extra work might be required from already-busy reviewers, and that reviewers might not need to alternate between critical reading and proofreading mindsets because they are not asked to suggest corrections for errors, only to detect and report them. Being asked beforehand to mark any noticed errors likely would incur *some* cognitive overhead, but we speculate that it would be less demanding than actual proofreading if the reporting mechanisms were very lightweight and did not involve much cognitive effort.

3.2.3 A model of low-level writing error detection

Similar to heuristic evaluations or usability testing [68], the noticing of low-level writing errors appears to meet the assumptions of a Poisson process: the finding of any given error by one reviewer is independent of whether it has been found by another reviewer because, in most cases, reviews are completed independently, and errors tend to be noticed randomly throughout a reading session. There are some complications with this model. For example, finding one error may lead to finding similar errors nearby, but we postulate that for the most

part errors in a paper are independent and thus their discovery is also independent. Even though each error may have a different probability of being found, we can still model the finding of the set of all writing errors in a paper as a single process because Poisson processes have an additive property—the rate of a combined process is the sum of the original two processes.

Just as heuristic evaluations yield different usability problem detection rates for different interfaces [68], we can expect that different proportions of errors will be found in different papers due to the unique characteristics of each paper. If we have λ, the average proportion of errors found by a single reviewer, we can use the following formula, adapted from Neilsen and Landauer's model for finding interface problems with heuristic evaluation [68], to calculate the proportion of writing errors we can expect to find with i reviewers.

$$ProportionFound(i) = 1 - (1 - \lambda)^i$$

In traditional descriptions of Poisson processes, λ refers to a count of occurrences, however here we use it as the proportion of total errors found, as did Neilsen and Landauer [68]. If we analyse the error detection rates of different papers across a set of reviewers to determine the λ for each paper, we can estimate how many reviewers might be needed to find a certain proportion of errors in a typical paper by using the formula and solving for i, the number of reviewers.

3.2.4 A mechanism for flagging errors

If the task of error noticing is to be incorporated into peer review by asking readers to flag errors they notice, then it is crucial to consider the flagging interaction. If the interaction distracts reviewers from their critical reading task, or simply is not fast enough, then the

additional time added to the existing reviewing task could be unacceptable, and it could reduce the quality of the reviews if too much cognitive effort is required for flagging. Because of this, the method of flagging errors needs to be simple and fast and it must minimize any interruptions to the flow of a reviewer's reading. The flag marked in the manuscript must also be expressive enough that an author can see where the error is and how the error might be fixed. With these requirements in mind, a simple text highlighting system seems like a suitable approach that could work with existing reviewing systems.

Highlighting text (digitally or on paper) is a common way of annotating documents. Picking up a physical highlighter to directly mark a section of text on paper is natural and straightforward; there is no confusion regarding the mapping from gestures to highlight marks, or the nature of feedback to authors. Digital highlighting is not as natural. A user must typically select a button to enter a highlight mode, which may be in a toolbar off of the document, or right-click a selection of text to open up a context menu with an option for creating a highlight. Without proper consideration for maintaining the reviewer's flow of reading, different interactions for creating digital highlights have the potential to be a major distraction during reviewing. To be prudent, we opt for a "no-menu" approach [25] to highlighting to minimize distractions. Our goal is to make digital highlighting as natural as highlighting on paper with a physical highlighting pen. We thus adopt a "modeless" approach in which the only action (gesture) a reviewer can take within the manuscript is to highlight selected portions that contain low-level errors.

Highlights provide a tradeoff between simplicity and expressivity. Highlighting errors would be simple for reviewers, but authors also need to understand what the errors are if this

flagging method is to be effective. Highlights are explicit in that they describe a precise location or range in which authors can locate the error, but they are tacit in that they are simply marks that are open to interpretation and may not immediately reveal the type or nature of the error. This tradeoff between simplicity and expressivity takes the burden off reviewers to describe flagged errors but transfers the responsibility of deciphering the nature of the error onto authors. This seems to be fair: reviewers can offer assistance by pointing out errors, but authors will be expected to identify and fix them. We postulate that for the most part, a highlight containing erroneous text is enough information for an author to deduce the nature of the error; for ambiguous annotations, an author could consult fellow authors or colleagues for a second opinion.

Having selected a technique for finding errors (error noticing) and for flagging them (highlighting) we explore in the next chapter whether this is a useful technique to include in peer review, by studying how effective it is at finding errors, and determining whether reviewers are likely to find it an acceptable amount of added work given the perceived benefit it provides. In the next two sections, we describe the materials that we developed and the prototype we implemented for use in the study.

3.3 Materials for Testing Error Detection Performance

Having more closely examined the problem of low-level writing errors in published academic papers, and having devised a low-effort technique that could mitigate the amount of errors that make it through to publication, the next logical step is to test whether our noticing and flagging technique can be a viable alternative to proofreading. In any experiment testing low-level writing error detection, we must first obtain a suitable set of

paper materials with errors to give the study participants. However, obtaining these materials is not straightforward. In this section, we discuss considerations for preparing such materials while attempting to preserve ecological validity.

3.3.1 Generating papers with errors

To test the error noticing capabilities of reviewers, it would be ideal to have a collection of real manuscripts that have been submitted for review, but not yet reviewed, with detailed knowledge of every writing error occurrence within the papers, for use as experimental materials. Obtaining such a collection of manuscripts would require either a large number of authors to supply earlier drafts of their papers, or some coordinated agreement between a conference or journal venue and its submitters, perhaps by asking submitters to opt in to share a copy of their originally submitted draft.

A secondary option for acquiring paper materials is to use already-published versions of papers and search for errors. Regardless of how the papers are obtained, the proofreading effort required to catalogue each error instance in each paper would be substantial. Furthermore, attempting to find enough papers to satisfy certain control conditions for an experiment (e.g., the count of errors in the paper or the frequency of certain types of errors) seems untenable without longer-term and coordinated efforts. Automated approaches could help with locating certain errors, but only if the experiment was predominantly concerned with the types of errors that can be detected by these algorithms, which tend to be limited to spelling and syntactical errors.

An alternative solution is to use existing published papers and manually insert errors into them, allowing us to control for both the frequency and types of errors. This approach is

convenient, but there are complications. Artificially introducing errors into a paper requires considerations for maintaining ecological validity—inserted errors should be representative of the types and frequencies of errors that we see in real papers.

Unfortunately, the research literature on the types of errors seen in published academic literature only provides us a limited picture: we know that misspellings [59] and errors in bibliographic citations [60] are common, but there is no work to our knowledge that analyses other types of errors in academic writing. There is a body of work, however, that looks at the types of errors seen in ESL contexts. For example, some studies have examined errors commonly made by Japanese or Cantonese speakers [69–71]. These analyses often result in distributions of error types. How representative these distributions are of the distributions of errors in published academic papers is unclear. We assume there are some significant differences. But without further data, the best we can do for now is to take these distributions and adjust them to what seems reasonable for academic papers, based on our own experience reviewing (and writing) papers that have writing errors in them.

Once we have a set of types of errors (a taxonomy) along with a reasonable distribution for the error types, we can begin inserting errors into papers. For each error type we draw from the distribution, there are two main considerations: what form will the error take and where will it appear in the paper. We chose the following approach: choose a random area of text and try to "massage" the error into the sentence, heading, or figure text in a way that seems as if it could reasonably have been an error, perhaps drawn or adapted from a set of real error examples of the same type of error.

The particular taxonomy of errors we use for error generation largely depends on what kind of writing we wish to simulate. For example, to be representative of the types of errors commonly made by ESL writers, we could include categories of errors that this is demographic prone to making, such as “Overuse of affixes,” “Duplicated comparatives or superlatives,” “Incorrect order of adverbials or adverbs” etc. [70]. In the case of academic writing, where little is known about the frequency of different types of errors, it is probably best to start with a more general and encompassing taxonomy with categories such as “Misspelling,” “Sentence Structure,” and “Punctuation” until more research and data collection has been done on the errors that occur in academic papers. As an initial step, we developed such a general-use taxonomy for generating errors.

3.3.2 A taxonomy of writing errors for academic papers

To insert errors into academic papers in a more ecologically valid way, we developed our own taxonomy of errors from which to draw examples (Table 3.1). The taxonomy consists of nine error types ranging for low-level misspellings to higher-level issues such as invalid sentence structure. Some of the lower-level errors such as article and verb form errors were adapted from a study on ESL errors [71], while the remaining categories were added to provide more complete coverage of the error types we had previously seen in our own and our colleagues’ papers.

Table 3.1: The error taxonomy we designed for choosing errors to manually insert into papers, and for providing a simple categorization scheme for reviewers.

Error Type	***Description***	***Example***
Misspelling	Spelling mistake or typographical slip, including missing or added letters, or duplicate words	*I had th cake we made yesterday for breakfast.*
Word misuse	Homophone confusion, malapropism, or incorrect usage of a word	*There methodology is highly sophisticated.*
Article error	Incorrect usage (or lack thereof) of the three articles: *a*, *an*, or *the*	*An banana is actually a type of berry.*
Verb tense or pronoun error	An error with the form of verb, pronoun, or agreement between the two	*I have seen him yesterday.* *Your selection of wines all tastes excellent.*
Possessive or plural error	Errors related to the plural form of a word or the use of apostrophes to show possession	*The population of octopi in the Pacific is steadily increasing. [octopodes or octopuses]* *Todays' sushi is half-price!*
Punctuation	Missing or improper use of punctuation marks, such as separating dependent clauses with a semicolon	*The door opened; because the man pushed it. [2nd clause not independent]*
Sentence structure	Incomplete or poorly structured sentences, missing a verb, or sentence splicing	*The humid air today in Vancouver. [not a clause]*
Overly informal Writing	Writing uses contractions, informal abbreviations, colloquial phrases or clichés, is written in the first person	*These results can't explain the discrepancies.*
Formatting	Any issues related to the appearance or structure of the paper: extraneous spacing, inconsistent header fonts or sizes, mislabeled figures or section numbers, etc.	*There is no extraneous space in this sentence.*

We had two main requirements when designing this taxonomy: (1) it should provide broad coverage of the types of errors seen in papers, and (2) it should be communicable to study participants (reviewers). The first requirement was to ensure we encapsulated the full range of errors that we might expect to see in an academic paper. In this sense, we took a top-down approach, looking at high-level error types and organizing them into categories. This is in contrast to in-depth taxonomies seen in the linguistics literature, which attempt to classify errors from the bottom up [72]. The second requirement (simplicity) is important so that we can elicit feedback on our taxonomy from participants. Most people are not familiar with the

multi-level and highly specific terminology used to describe errors in the linguistics literature. Because our taxonomy is experimental, we wanted to able to have a dialogue about it with potential study participants.

3.4 A Simple Reviewing Tool with Error Flagging

In order to test how well reviewers can detect errors in academic papers, we designed a web-based prototype to support reading and flagging errors in papers using a highlight-based annotation mechanism. We implemented the prototype using HTML/CSS/JavaScript because of the relative ease of creating highlights using the `<mark>` tag and we used a simple web server to provide HTML representations of papers. The interface is split into two main components: the Document view, where the user can read and annotate a paper, and the Reviewing view, where the user can rate a paper on various dimensions of quality. The user can switch between the two views as often as desired before clicking a button to submit the review.

3.4.1 Document view

The document view (Figure 3.1) provides a traditional vertical scrolling view of a paper. The user can create, select, or delete highlights over any contiguous segments of text on a page. An issue counter in the top-right counts the number of highlights (issues) in the document—clicking the adjacent up or down arrows cycles through the created highlights and selects the next or previous highlight, centering the viewport on that highlight's coordinates. The up and down keyboard keys perform the same cycling operation. A secondary feature allows selected highlights to be classified and colour-coded into predefined categories (discussed below). The review button in the top-right corner places the user in the reviewing view.

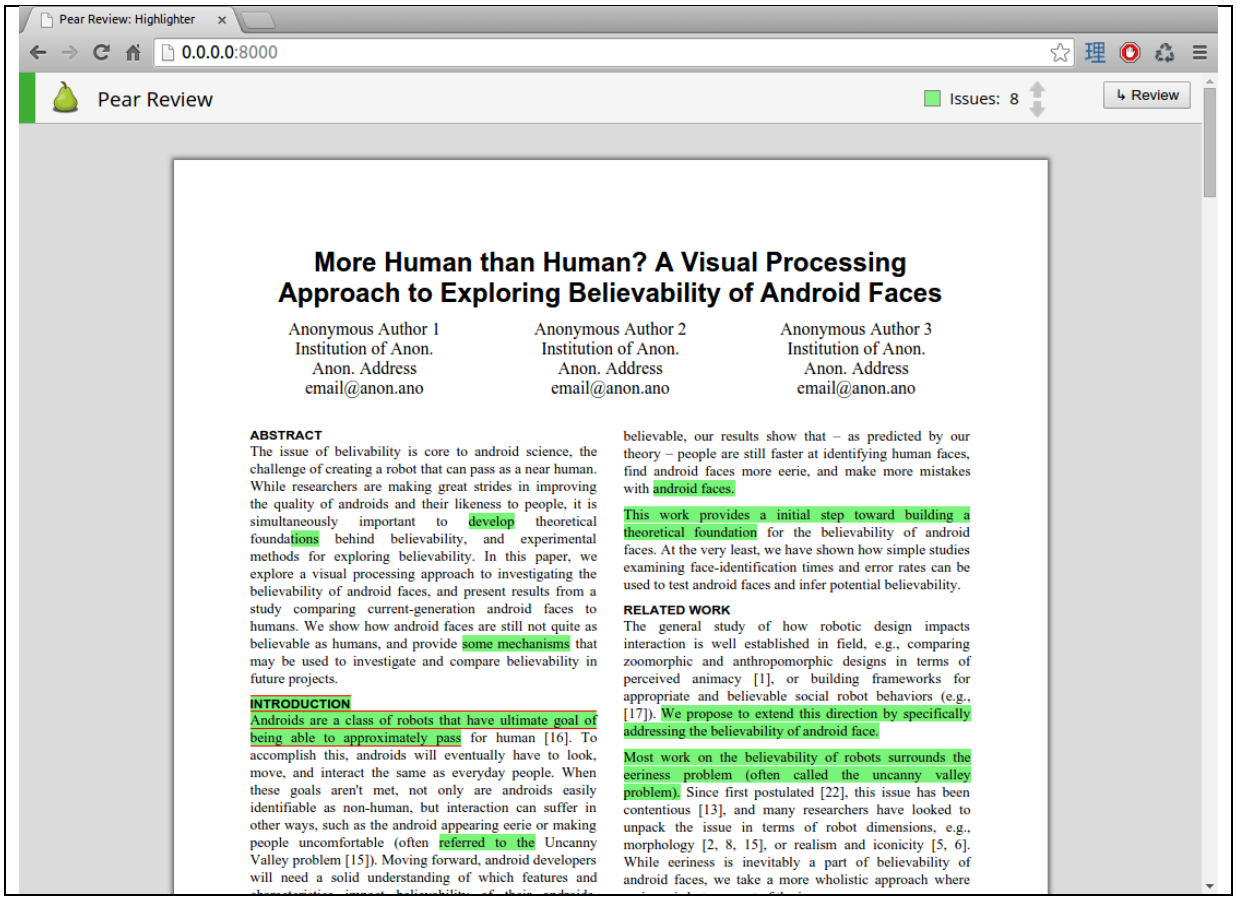


Figure 3.1: The document view in our web-based prototype, where participants can highlight arbitrary contiguous segments of text. The currently selected highlight is outlined in a red border.

HTML offers no native support for paginated documents. To simulate the look and feel of conference-style papers typically seen in computer science, we created an HTML/CSS template with a double-column paper format. Due to the lack of pagination support, papers imported into the reviewing tool had to be manually broken up into pages.

Converting papers from a PDF to our HTML takes a considerable amount of manual work: copying and pasting text from PDFs often results in spacing and text artefacts that need to be manually fixed. We could find no automated tools that were sufficient for converting PDF papers to HTML, nor any that supported pagination.

In order to highlight a segment of text using our reviewing tool, the user clicks and drags the mouse to create a text selection and presses the spacebar to create a highlight. Upon creation, a highlight becomes *selected*—indicated by a red border along the top and bottom of each row of text. Any highlight will become the selected highlight when clicked with the mouse pointer, deselecting all other highlights. A highlight can be deleted by selecting it then pressing the delete key. Highlights can be as small as a single character, or can span several paragraphs or other DOM elements. We arbitrarily chose a light shade of green for the highlights because we felt this was more pleasing to the eye than the default neon yellow used by our browser.

Our choice of highlighting interaction (text drag to select plus spacebar to highlight) was based on a couple of factors: we wanted a simple, no-menus approach [25] that did not distract the reviewer from the reading task, and we wanted the interaction to be simple and quick in order to reduce the perceived effort by busy reviewers. Discussions among our colleagues led to two simple interactions: click-and-drag to highlight, versus the same interaction plus an additional spacebar click to highlight. We noted that some people like to "busy-select," i.e., play with the cursor and select text while reading, and we reasoned that it would be irritating if every selection resulted in a highlight, so we decided on the space bar method to highlight the currently selected text instead of the simpler click-and-drag.

In addition to a basic highlighting mechanism, the tool also supports the categorization of highlights. If categorization mode is enabled, a popup (Figure 3.2) appears on the margins of the page whenever the user selects a highlight. This popup is vertically aligned with the highlight and provides a number of colour-coded error categories from which to choose; the

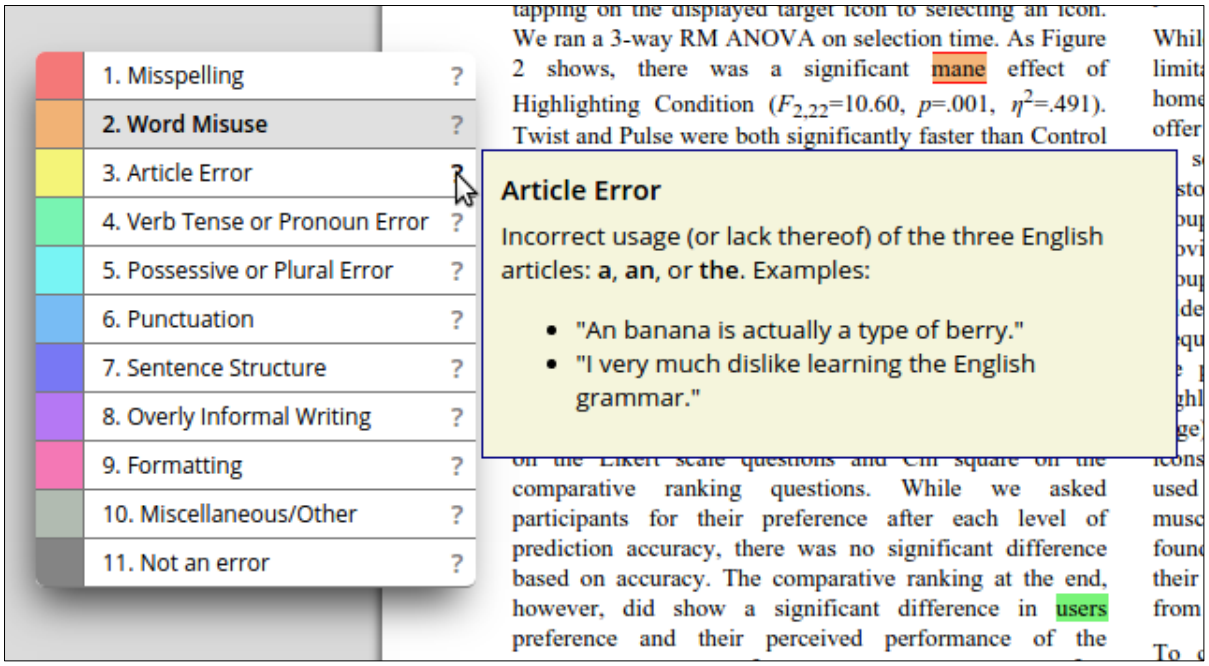


Figure 3.2: In categorization mode, the classification box appears in the margins next to a selected highlight. Choosing a category re-colours the highlight. Hovering over the "?" icon reveals a description of the error type.

categories correspond to the error taxonomy we previously described. Choosing a category changes the highlight to the corresponding colour. Help text is accessible for each category by hovering the mouse over a question mark icon—the hover text provides a description of the error type along with examples. In addition to the issue counter and cycle buttons, when categorization mode is enabled, a second counter appears in the top bar; it counts and cycles through the highlights that have yet to be classified. All unclassified highlights use the standard green colour by default.

At the DOM level, we used HTML5 `<mark>` elements that surround text nodes in order to create highlights. JavaScript functions make it easy to acquire the DOM range object for a user's text selection and surround it with the mark element. However, when a text selection spans the text from multiple HTML elements, for example, a selection starting in one

paragraph element and ending in the following paragraph, creating the highlight is slightly more complicated because surrounding the selection with a mark element will break the tree structure of the DOM. To solve this problem, we generate a list of all the text nodes from each element comprising the text selection and create mark elements around each of those nodes. In this case, a single highlight will have multiple mark elements, but we treat them as a single highlight by assigning an identification number in a custom data attribute in each mark element. This way, when a user selects or deletes one of the marks, they are all selected or deleted. For a quick overview of how DOM highlighting works, see Appendix B. The described marking implementation also works for overlapping highlights, but it does not directly provide a suitable visualization mechanism—a topic discussed further in Chapter 5.

3.4.2 Reviewing view

The reviewing view is where the user completes their subjective review of a paper (Figure 3.3). They can switch between this view and the document view at any time during the review session by clicking the button in the upper-right corner. The review consists of 12 Likert items that gauge the user's opinion on the potential impact of the research, the methodology used in the research, the writing quality of the paper, and how qualified they feel they are to review the paper. The bottom of the review contains a text-box for open-ended general comments. A full list of the Likert items can be found in Appendix C.1. Once the review is complete, the user can press the "Submit Review" button at the bottom of the page.

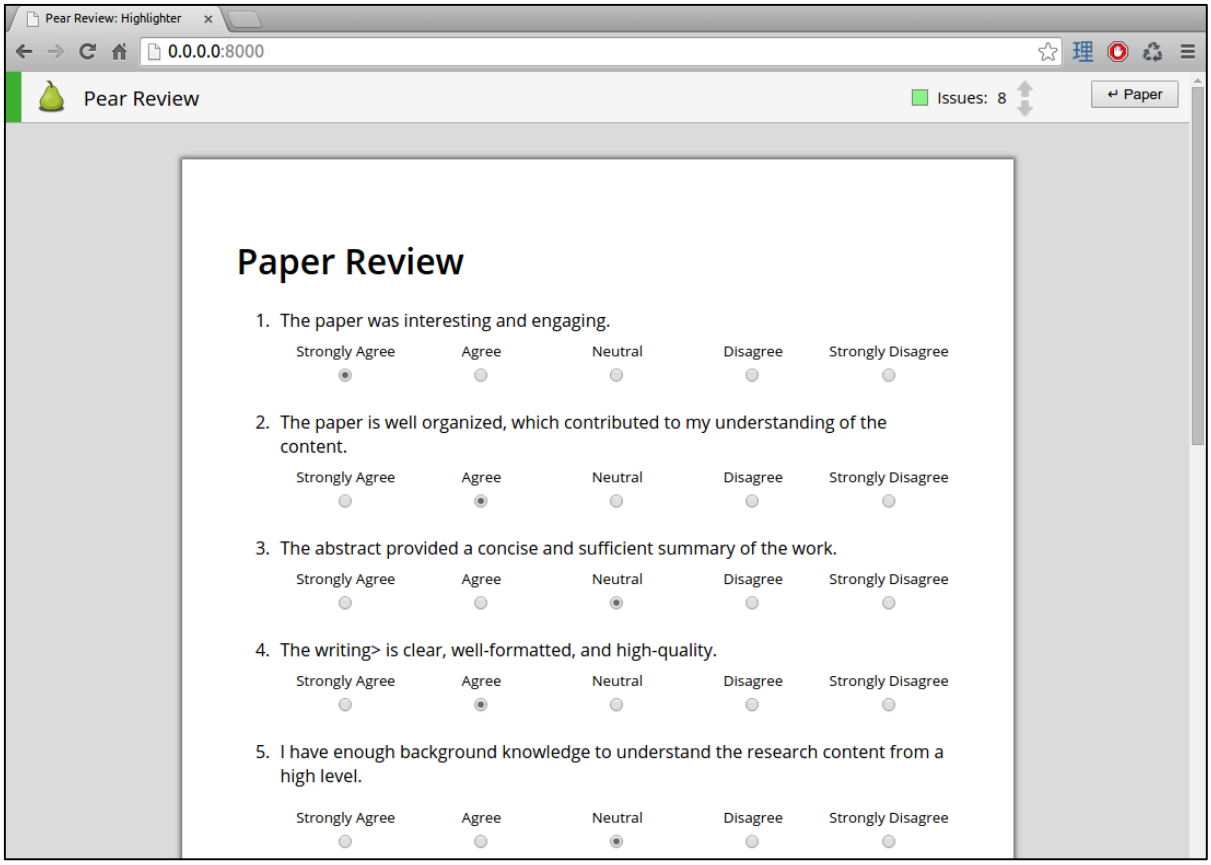


Figure 3.3: The reviewing view provides a simplified review form with a set of radio buttons for various levels of quality, along with a textbox for open comments at the end.

3.4.3 Annotation storage and retrieval

In order to analyse the errors that participants find, we need the ability to store and reload their highlight annotations. After a participant presses the submit button in the reviewing view, the application generates a JSON file containing all of the highlights. More specifically, the file stores a set of child element indices representing the starting and ending elements that the highlight intersects, relative to the root DOM element. In addition to the element offsets, we store two more indices that represent the textual offsets within the element for the start and end of a highlight, along with extra meta-data such as timestamps, whether the highlight has been deleted, and how it was categorized (if at all).

Annotations can be loaded from a JSON file back onto the original paper by following the child offsets from the root element and re-inserting marks at the appropriate text offsets. One crucial implementation consideration is to ensure that element and text offsets are stored such that they are independent of other highlights. This ensures that the child index offsets are not dependent on existing highlight mark elements, allowing them to be reloaded in any order and allowing for multiple sets of highlights to be loaded at the same time so that reviewers and authors can view all of the errors detected in a paper, possibly by multiple reviewers.

3.4.4 Related work in web annotations and highlighting mechanisms

Using the browser to implement research prototypes involving web annotations is not new. A number of other researchers have taken this approach [25,27,31,33,34,41,73,74]. A great deal of work has been done not only on implementing annotations on a webpage within a browser, but also on how annotations can be distributed or shared across the World Wide Web. Nevertheless, there are still a lack of different standards and protocols for doing so [36]. Some organized efforts such as the W3C Web Annotation Working Group are underway to standardize annotation representation, storage, and accessibility across the web [75]. The central research focus here is on how annotations can be stored and retrieved separately from the base document (which may contain copyrighted material) [25], how they can be abstracted away from a specific document format [25,38], and how annotations should be repositioned or orphaned in a changing document [33,73]. One application directly related to peer review is the implementation of a set of reading tools for use in the Open Journal Systems [31].

Highlighting is one of the most common and simple digital annotation features, but a number of design and implementation considerations can arise: for example, how to create the highlight, or how to disambiguate overlapping highlights [31]. To create highlights, most document authoring and document reading software, including a number of research prototypes, uses a simple click and drag interaction to select text with a separate button to highlight the selection, sometimes via a right-click context menu. One method commonly used to visualize overlapping highlights is to make overlapping regions a darker shade of colour [74].

3.5 Summary

The error noticing and flagging tool we developed is a prototype. It is intended to be a minimally intrusive interface that supports lightweight annotation of academic papers through simple highlighting, with the additional option of assigning colour-coded categories to highlighted text. In the next chapter, we describe a controlled laboratory experiment that assesses the prototype in terms of our goals. The experiment also serves to partially validate our Poisson distribution assumption about error detection and provides an estimate for the number of reviewers that might be required to reliably detect a desired percentage of the errors present in a document.

Chapter 4: A Study on Catching Writing Errors

In this chapter, we present a preliminary investigation into how we can use computer interfaces to improve the quality of writing at the level of minor low-level writing errors (correct spelling and grammar, and elimination of formatting errors) by augmenting existing workflow in the peer review process to include support for flagging low-level writing errors. Adopting the idea of "error noticing" from Chapter 3, we conducted a formal laboratory study using prototype software to test whether simple "highlighting" of low-level errors noticed during review would be useful (in terms of identifying significant percentages of actual errors) and acceptable (in terms of it being likely that reviewers will be willing to use the software). The results from the study are promising. Having three or more reviewers seems to be adequate to detect 80% of the errors in the test documents if reviewers are properly primed about the task, and participants in the study indicated a willingness to use software that supports highlighting low-level errors during the reviewing process.

4.1 Method

We conducted a study that measured how effective reviewers are at noticing and flagging low-level writing errors during a peer-reviewing task for papers chosen from the literature, along with their subjective opinions on the usefulness and acceptability of including such processes in real peer review sessions. A key variable in the experiment was whether participants were primed with an initial error-classification task before completing their reviews. In order to capture error noticing in an ecologically valid way without participants resorting to proofreading, we included deceptive elements in the experiment to delude the

participants into believing that they were performing a legitimate peer reviewing task on the test papers that they were given.

4.1.1 Participants

Our participant pool consisted mostly of graduate students in computer science, engineering, and library sciences at the University of British Columbia. Participants were required to have native English speaking ability, and to not be colour blind because our interface uses a range of colours for highlighting text. We chose to recruit graduate students because they were more likely to be at least familiar with the academic peer review process than undergraduates or the general population. Faculty researchers were also invited to participate, but we were unable to recruit any due to their busy schedules.

We required that participants be from fields related to Human Computer Interaction (HCI)—computer science, engineering, and information science—so that the content of our chosen set of test papers would be more accessible and familiar to them. Our experiment could have been run with participants from any field of study, but the need to select a set of test papers necessarily limited the pool of reviewers from which we could recruit. Although noticing writing errors in a paper does not require expertise in any particular field, we had to convince participants that they were actually performing a reviewing task; recruiting participants from unrelated fields to review HCI papers would likely have aroused suspicion about the true nature of the experiment.

We recruited 24 participants (9 female, 15 male) from the graduate student population in engineering, computer science, and library sciences at UBC; they consisted of 23 graduate students and 1 undergraduate (male) who had taken graduate-level courses. The participants

had relatively little experience reviewing papers for conferences or journals: 16 participants had never formally reviewed a paper, 6 participants had reviewed one to five papers, and 2 participants had reviewed over sixteen papers. Each participants was compensated $20 CAD for their time.

4.1.2 Apparatus and materials

We selected a set of published papers from the field of HCI and prepared them for the reviewing task by converting them into a format readable by our annotation tool and inserting errors into them. We created a simplified report form to be completed after the reviews, along with a post-test questionnaire (Appendix C.1).

Papers

We chose four published short (4-5 page) HCI papers to use in the reviewing task in the experiment, and one unpublished paper for the classification task. We specifically chose HCI papers in order to widen our participant pool: HCI is a very multidisciplinary field so papers tend to be more accessible to those with non-computer-science backgrounds. One of the papers was published in the proceedings of The International Conference on Tangible, Embedded, and Embodied Interaction (TEI), 2011. Two papers were published in the proceedings of The SIGCHI Conference on Human Factors in Computing Systems (CHI), 2013. The fourth paper was published in the proceedings of The International Conference on Human-Agent Interaction (iHAI), 2014.

We manually converted each paper into an HTML representation for viewing and annotating in our web interface, preserving the look and pagination as closely as possible. Following Gould and Grischkowsky [62], we carefully crafted and inserted a set of writing errors into

the papers according to the error generation methodology described in Section 3.3.1. We selected error types from the taxonomy we developed and came up with examples that would fit in various locations in the paper. In the cases where we discovered already existing errors in the papers, we elected to use those instead of generated errors. We explicitly coded each error along with its corrected version as HTML element pairs, one of which was always hidden from view. Adding errors as HTML entities allowed us to generate papers with different numbers of errors by either disabling or enabling a corrected version of each error instead of the error itself. The error HTML elements also made it easier to identify when a participant had found an error by enabling us to visually identify overlaps with the highlights they created.

Each paper came in two versions: *high-error* and *low-error*. The high-error versions had 40 errors each, while the low-error versions had 20 errors each. The low-error versions contained a subset of the errors from the high-error versions of a paper, but were chosen such that the relative frequency of the error types remained the same. For example, if the high-error paper contained 25% misspellings, the low-error version also contained roughly 25% misspellings. We chose to use two different versions of each paper in order to test whether the frequency of errors had an effect on error noticing performance. The low-error version of a paper was simply the high-error version with the selected half of its errors disabled.

The four published papers we chose for the reviewing task were code-named *AndroidFaces*, *HapticDesign, Notifications*, and *Password*. The paper for the classification task was code-named *Ephemeral*; it was obtained from a colleague and was undergoing review at the time of the study. *Ephemeral, Notifications*, and *Password* described user studies measuring

performance on various computer interfaces, *AndroidFaces* described a social human-robot interaction study, and *HapticDesign* offered guidelines for designing haptic interfaces. Because of the lack of literature on the distribution of error types in academic writing, and because we know misspellings are common [59], we devised a custom error type distribution that seemed reasonable to us for inserting errors. Figure 4.1 shows the distributions of inserted error types in each paper. An example of one low-error paper with the inserted errors highlighted is provided in Appendix C.2.

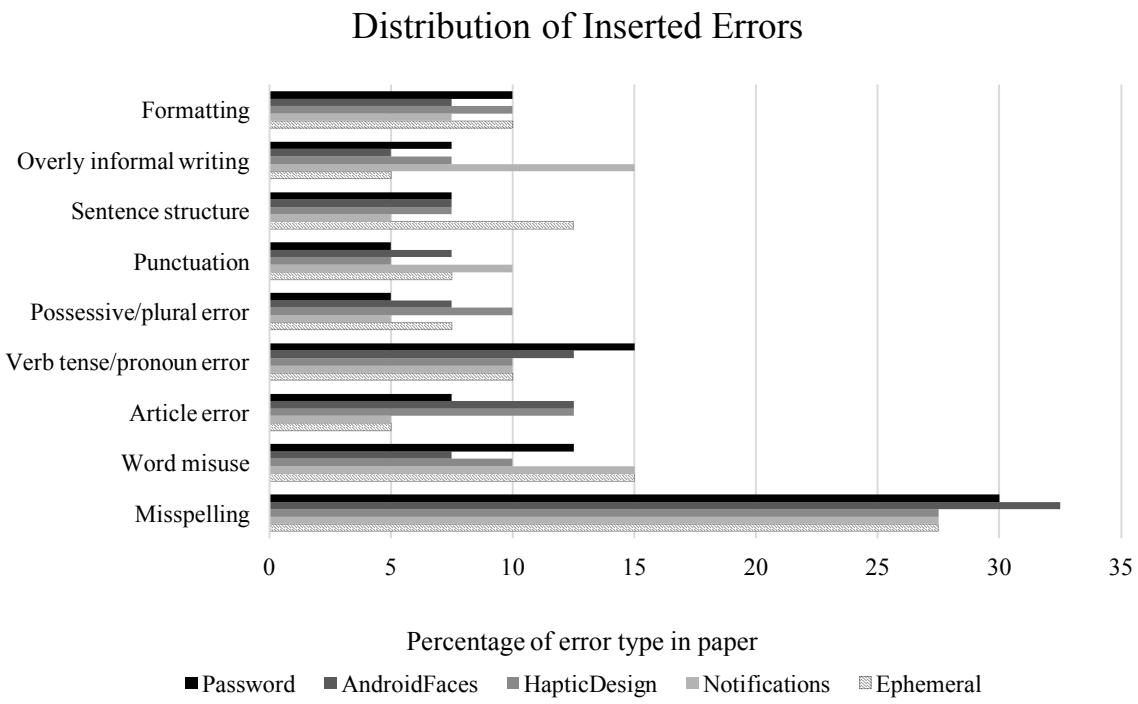


Figure 4.1: Distribution of error types in each paper by percent. The high- and low-error versions used the same distribution.

Review form

The review form consisted of twelve Likert items that probed participants' subjective opinions on a paper over various dimensions of quality, along with a textbox at the end for additional comments. Although real reports used in peer review typically involve a series of written components, we elected to use this more simplistic style of review for two key reasons: having fewer written questions allowed participants to complete two reviews in the allotted time, and having a simplified review played an important role in our deceptive narrative. This deception and the reasons why it was necessary are described in detail in Section 4.1.4. The full list of questions included in the review is provided in Appendix C.1.

Questionnaire

We designed a post-test questionnaire to elicit participants' subjective opinions on their willingness to perform error noticing while reviewing, its overall usefulness of the error noticing idea, and their experiences using the annotation tool. The questionnaire consisted of demographics questions, eight Likert items, and a single open-ended question for general comments. The full questionnaire is provided in Appendix C.5.

Workstation

The tasks were performed on a desktop computer running Windows 8 with the annotation interface running in Google Chrome. Participants viewed the interface through a widescreen monitor (2560 x 1440 resolution) and the default zoom of the browser was such that they could see approximately one full page of a paper at a time.

4.1.3 Tasks

Participants completed two tasks: a paper reviewing activity and a classification activity. The participants were randomly assigned to one of two equal-size groups. Participants in the primed group first performed the classification activity followed by the paper reviewing activity. Unprimed participants completed the two tasks in the opposite order.

Reviewing Task

In the reviewing task, participants were given two papers to read and review using our reviewing and annotation software prototype that was described in Section 3.4. They were asked to highlight any low-level writing errors they encountered while reviewing and to complete a review report using a form that was provided when they were ready. After the first paper review was completed, the prototype enforced a one-minute break before loading the second paper.

Classification Task

In the classification task, participants were given a paper with a set of pre-highlighted writing errors. They were asked to look at each highlighted error and determine what type of error it was. Classification was done by clicking on the highlight to show a classification popup menu (Figure 3.2) and selecting a category corresponding to the error taxonomy discussed in Section 3.3.2. Two additional categories were present: “Miscellaneous/Other” and “Not an error,” in case participants felt that an error did not fall into any one of our pre-defined categories or felt that it was not an error at all. Participants were not required to carefully read the classification paper. Examining the surrounding context of each highlight was

intended to be enough to correctly classify the error. This was explained to the participants. Each participant received the same paper for the classification activity.

The classification task served three purposes. First, for the *primed* group that performed the classification task first, it allowed us to see whether an initial task in reasoning about how to classify errors would have an effect on their error noticing performance in the reviewing task that followed. We postulated that this task would set their expectations for what we considered to be a low-level error and would thus increase the number of errors they found.

Second, the classification task allowed us to measure the reliability of our writing error taxonomy by measuring agreement with our internal classifications (each highlight had meta-data, hidden from the participants, indicating which category we believed it to be in). Although the classification task would not have an effect on unprimed participants who performed the classification task *after* the reviewing task, we still had them complete the task in order to gather additional data on agreement between participants' classification and our internal classification.

Third, the classification task could give us a picture about how participants feel about the classification interaction itself, and whether it might be useful (or overly tedious) to include as a feature in future reviewing interfaces. We also wanted to test our assumption that adding more work for reviewers—by having them classifying errors *and* highlighting errors—would make them less willing to provide writing assistance during peer review.

4.1.4 Procedure

Each participant was randomly assigned to one of two experimental groups: *primed* or *unprimed*. Each group contained 12 participants. Those in the primed group first performed

the classification task, followed by the reviewing task in which they reviewed two papers. Conversely, participants in the unprimed group first reviewed two papers and then performed classification. Each participant was given one high-error version of a paper, and one low-error version of a different paper. In total we had eight paper-versions—four papers each with high-error (40) version and a low-error (20) version—counterbalanced such that each experimental group saw all twelve orderings of two papers chosen from four papers, but with the opposite high-low patterns. After the experiment, participants were debriefed and asked to complete the post-test questionnaire. The experiment took approximately 1 to 1.5 hours. Participants received a $20 CAD honorarium for their time. The experiment was approved by the UBC Behavioural Research Ethics Board and we received permission to use deception in the experiment.

After participants arrived, they were seated in a room with a computer desk. The experimenter told the participant the following story (a falsehood) about the purpose of the experiment, explaining that the papers being reviewed would be used as materials in an upcoming course for teaching graduate students how to do peer review. It was explained that the reviews would help the course organizers filter a variety of weak and strong papers to be included as course materials.

> *"We are currently designing a course that's meant to teach new grad students in computer science how to do effective peer review, but first we need decide on which materials will be used for weekly readings and assignments. We have gathered a large collection of candidate papers and we would like you to use our tool to rate them based on how suitable you feel they are for the course, and give your subjective opinion on various dimensions of quality. Once we have gathered enough responses from people, we will be able to filter which papers we want to use more easily, with some strong in*

some areas and weak in other areas, to facilitate learning. Another purpose of this experiment is that we would like to test out the software we designed, since this rating and filtering tool could be used for other fields as well."

The experimenter then introduced the participant to the paper reviewing and annotation tool. For participants in the primed group, the experimenter explained that he wanted to first test out the classification "side feature," and took care to downplay its relevance to the overall task the participant was being asked to perform.

"Before we begin the main task, I would like you to run through one of the side-features we are testing. Here you can see there is a paper with a bunch of highlights. Each highlight indicates that there is some kind of low-level writing error. If you click on one, you'll see that this classification box pops up and you can choose which type of error you think it is. If you are not sure where it belongs, you can hover over the question mark to see examples of each type of error. I have also printed out the descriptions here if you prefer that. I would like you go through each of these errors and classify them. You do not need to really read the paper, just the context surrounding the errors."

After the priming task for those in the primed group, and as the first step for those in the unprimed group, the experimenter instructed the participant to review two papers chosen randomly by the software. In an effort to further convince a participant that we were primarily interested in the reviews (and not finding errors), the introduction to the task was carried out with the review screen initially opened up and visible. The experimenter reiterated that it was fine if a participant did not feel fully comfortable reviewing the paper, and that the subjective opinions would be helpful to us. Our intention in doing so was to reduce any discomfort (and suspicion) associated with a participant being asked to review a paper the participant did not feel qualified to review.

After describing the reviewing view in the prototype, the experimenter showed the participant the paper view. The experimenter explained that, because many of the papers we had acquired were rough drafts or were even unpublished, the drafts might contain some low-level writing errors such as spelling or grammar issues, and that we would like help in fixing the errors before the course began, since we did not want the presence of writing errors to skew future students' perceptions of the papers. The experimenter then showed the participant how to create highlights and asked the participant to mark any errors the participant *happened* to come across while reviewing the paper, emphasizing that we did not want the participant to spend time proofreading.

Participants were left alone in the room while they performed the classification and reviewing tasks (in either order). The experimenter re-entered the room between tasks to explain the second task. The tasks were performed using the desktop computer and software described earlier. Once participants completed both tasks, the experimenter re-entered the room and debriefed the participant about the real purpose of the experiment along with all elements of deception. Participants were then paid the honorarium and were asked to complete the post-test questionnaire. The questionnaire was administered *after* revealing the deception because it contained questions related to the highlighting activity which we had earlier deemphasized, and we wanted to get their opinions on the real error-finding task.

4.1.5 Hypotheses

We had five hypotheses about the outcome of the experiment. The numeric thresholds we chose were based on our pilot testing and extrapolations from what we read in the literature.

H1. Three primed participants will be able to collectively catch at least 75% of writing errors in a paper, surpassing Gould and Grischkowsky's single proofreader performance of 70% [62].

H2. There will be no difference in error detection rates between the high-error and low-error versions of the papers.

H3. Primed participants will detect more errors than unprimed participants.

H4. Reviewers will agree with our taxonomy at least 85% of the time.

H5. Reviewers will be willing to flag errors in real peer review sessions, but will be less willing to classify them.

4.2 Results

We first examined whether the data supported our assumption that error noticing was a Poisson process. We then examined whether priming, error frequency, or error type had an effect on percentage of errors noticed, after which we looked at how well our taxonomy of error types matched participants' classifications. Finally, we looked at participants' responses to questions about their willingness to use the prototype for real reviewing sessions.

4.2.1 Poisson model fit

The Poisson model described in Section 3.2.3 appears to fit the error noticing data quite well. Table 4.1 shows the average proportion of errors found by a single reviewer in each version of each paper, along with the fitted proportion in the model.

Table 4.1: Results of fitting the error-noticing data to our Poisson model. "L" indicates the low-error frequency of a paper and "H" indicates the high-error frequency. Note that the papers do not contain exactly 20 and 40 errors: this is because participants found existing errors in the papers that we had not detected. These errors were included in the analysis.

Paper	***Subjects per Paper***	***Total Known Errors***	***% of Errors found by one reviewer***	***Model***	
				Best fit λ	***R^2***
AndroidFaces (L)	6	*26*	0.25	0.22	0.970
AndroidFaces (H)	6	*46*	0.39	0.37	0.973
HapticDesign (L)	6	*26*	0.50	0.46	0.980
HapticDesign (H)	6	*46*	0.29	0.28	0.999
Notifications (L)	6	*22*	0.27	0.23	0.935
Notifications (H)	6	*42*	0.32	0.26	0.920
Password (L)	6	*26*	0.27	0.22	0.898
Password (H)	6	*46*	0.27	0.24	0.970

The fitted proportion appears to consistently underestimate the actual single-reviewer proportion by an average of about 3.5%. This may be because certain errors were more difficult or impossible for participants to find, and thus once the easier problems were found the rates of finding more errors dropped off. We conclude that for our purposes error noticing is well approximated by a Poisson process model

Figure 4.2 shows the fitted models and the actual data points from the experiment. To calculate each data point (the proportion of errors caught by X reiewers), we first calculated the proportions of unique errors caught by every combination of X reviewers, then averaged

those proportions to obtain a single data point. As with finding bugs in different usability problems, the λ value varied across the different papers, ranging from just over 0.25 to 0.50 for a single reviewer.

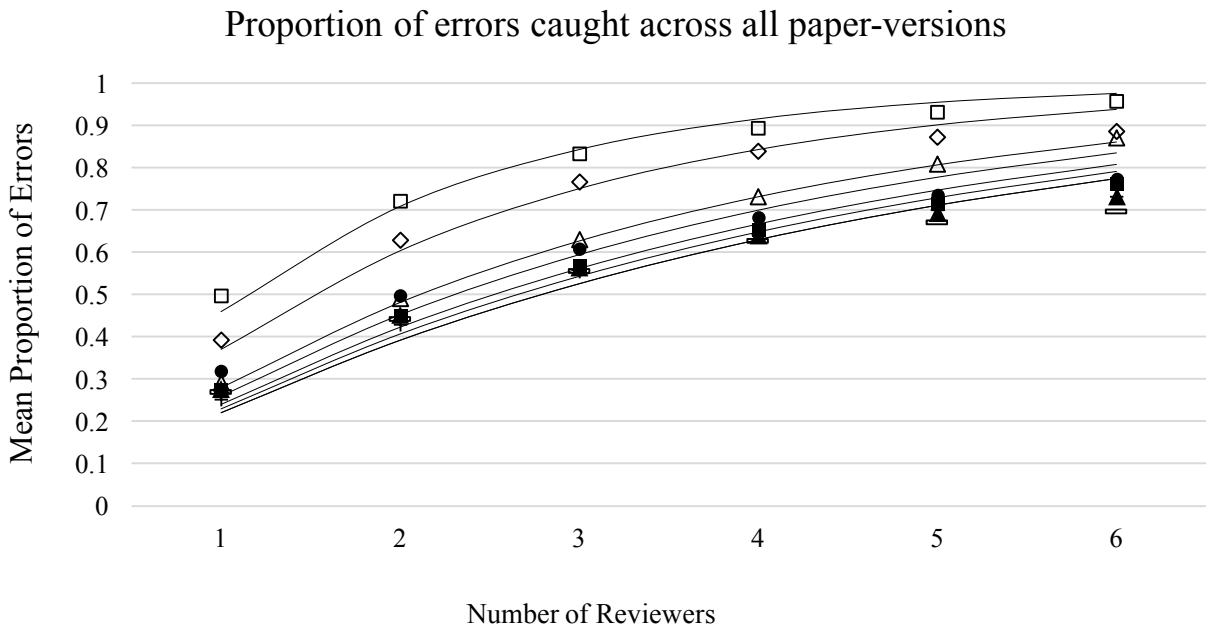


Figure 4.2: The average proportion of errors caught for every set of X reviewers (black and white markers) compared to the fitted Poisson model (line), for each of the eight paper-versions.

The average single-reviewer proportion of errors found across *all papers* for the primed group (λ=0.454) was higher than the unprimed group (λ=0.186). When adjusted down by 0.035 to better fit the model, these rates become (λ=0.419) and (λ=0.151) respectively. Plugging these values into our model shows that a group of three average reviewers reviewing the average paper can be expected to find 80.4% of all writing errors, exceeding our usefulness threshold of 75% and confirming **H1** (see Figure 4.3).

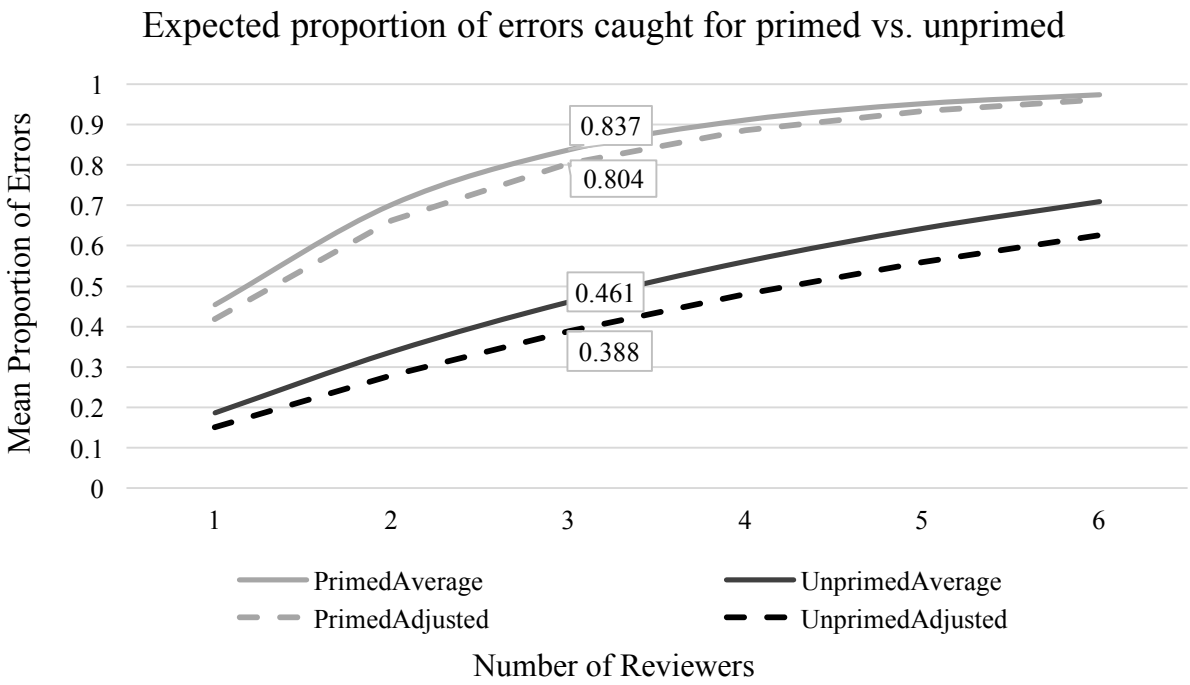


Figure 4.3: The expected proportion of errors a group primed and unprimed reviewers can expect to find in the average paper. The dashed lines use the adjusted single-reviewer rate to account for overestimation.

The curve showing the primed participants' performance appears to be much higher than the per-paper curves in Figure 4.3, and similarly the curve for unprimed participants' performance appears to be much lower—this is because the data for each paper contained a mixture of primed and unprimed participants. As a result, the curve for each paper is likely in the middle of where it would be had the participants all been primed or unprimed.

4.2.2 The effects of priming and error frequency

We ran a mixed-design analysis of variance (ANOVA) to measure the effect of priming and of error frequency on error noticing performance. The dependent variable, performance, was measured by calculating the proportion of errors found out of the total known errors in the paper. All effects met the assumption of equality of variances as tested by Levene's procedure, and the within-subjects factor met the assumption of sphericity as tested by

Mauchly's procedure. There were no significant interaction effects between priming and error frequency, nor was there a main effect of frequency (confirming **H2**). However, there was a statistically significant effect of priming, $F(1,22) = 18.98, p < .05, \eta^2 = .453$ (confirming **H3**). See Appendix C.6.1 for the full SPSS ANOVA output.

4.2.3 Reliability of our error taxonomy

All 24 participants classified 40 errors in the *Ephemeral* paper using categories from our error taxonomy. The agreement of classifications differed across participants, but on average was moderately high (M=31.3[78.3%], SD=3.9[9.8%]). In total, there were 960 classifications, 208 of which disagreed with our own. The agreement rate of 78.3% did not meet our expectation of 85% agreement (failing to confirm **H4**). Table 4.2 shows the percentage of disagreement by category, and Table 4.3 shows how the disagreements were distributed over the participants' classifications. In the table, we use percentages for comparison across categories, rather than counts, because each category had a different number of errors.

Table 4.2: Heat map showing the level of disagreement within each error category. For example, 60% of all "Sentence Structure" errors were classified as something else by participants. Note: the error types are truncated for space—for the full category names see Table 3.1.

	Spell	Word	Article	VTP	P/P	Punc	Struc	Infor	Form
Disagreement (%)	15.9%	23.6%	12.5%	13.5%	22.2%	20.8%	60.0%	2.1%	9.4%

Table 4.3: Heat map of the distribution of disagreements by category. The left headings represent our own classifications, and the top headings are how the participants classified the errors. For example, of all the disagreements where we labelled the error as "Misspelling," 54.8% were labelled as a "Word Usage" error.

	Spell	Word	Article	VTP	P/P	Punc	Struc	Infor	Form	Other	NotErr
Spell		54.8%	0.0%	2.4%	2.4%	0.0%	2.4%	0.0%	33.3%	4.8%	0.0%
Word	70.6%		0.0%	2.9%	14.7%	2.9%	2.9%	0.0%	0.0%	0.0%	5.9%
Article	0.0%	0.0%		0.0%	0.0%	0.0%	50.0%	0.0%	16.7%	33.3%	0.0%
VTP	38.5%	38.5%	0.0%		15.4%	0.0%	7.7%	0.0%	0.0%	0.0%	0.0%
P/P	18.8%	25.0%	0.0%	0.0%		43.8%	0.0%	12.5%	0.0%	0.0%	0.0%
Punc	0.0%	0.0%	0.0%	0.0%	0.0%		6.7%	6.7%	80.0%	6.7%	0.0%
Struc	1.4%	0.0%	1.4%	5.6%	0.0%	25.0%		15.3%	8.3%	33.3%	9.7%
Infor	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%	0.0%		0.0%	0.0%	0.0%
Form	0.0%	11.1%	0.0%	11.1%	0.0%	22.2%	11.1%	0.0%		11.1%	33.3%

Participants often misclassified errors that we placed in the "Sentence Structure" category: 60% of these instances were given a different classification by participants. "Word usage," "Possessive or plural error," and "Punctuation" were also often misclassified, with rates of disagreement at 23.6%, 22.2%, and 20.8% respectively. The remaining categories were misclassified less than 16% of the time, with "Informal Writing" being the least misclassified (2.1%).

Among "Sentence Structure" errors, most disagreements resulted in an "Other" (33.3%), "Punctuation" (25.0%), or "Informal" (15.3%) classification. Most "Word Usage" errors were overwhelmingly classified as "Misspelling" (70.6%), and most "Misspelling" error were classified as either "Word Usage" (54.8%) or "Formatting" (33.3%) errors. Most "Possessive or plural" errors were misclassified as "Punctuation" (43.8%), "Word Usage" (25.0%), or "Misspelling" (18.8%) issues, and "Punctuation" issues were overwhelmingly misclassified as "Formatting" (80.0%).

4.2.4 Subjective responses by participants

We gathered data from a set of eight Likert items asking participants about their subjective experiences during the experiment (Figure 4.4). All 24 participants agreed that the highlighting mechanism was easy to use. A minority of the participants (6) felt that error noticing was more time consuming than regular reviewing, but the majority felt that the highlighting mechanism was sufficient for pointing out writing errors (17). Most agreed that the error noticing and highlighting method would be useful to include in real peer review sessions (22), and they were willing to do it themselves (22). Fewer, but still a majority of participants, felt that classifying along with highlighting errors would be useful in peer review (15), and a similar amount were willing do so themselves (14), confirming **H5**. Just over half the participants agreed that they had highlighted every error they came across (15).

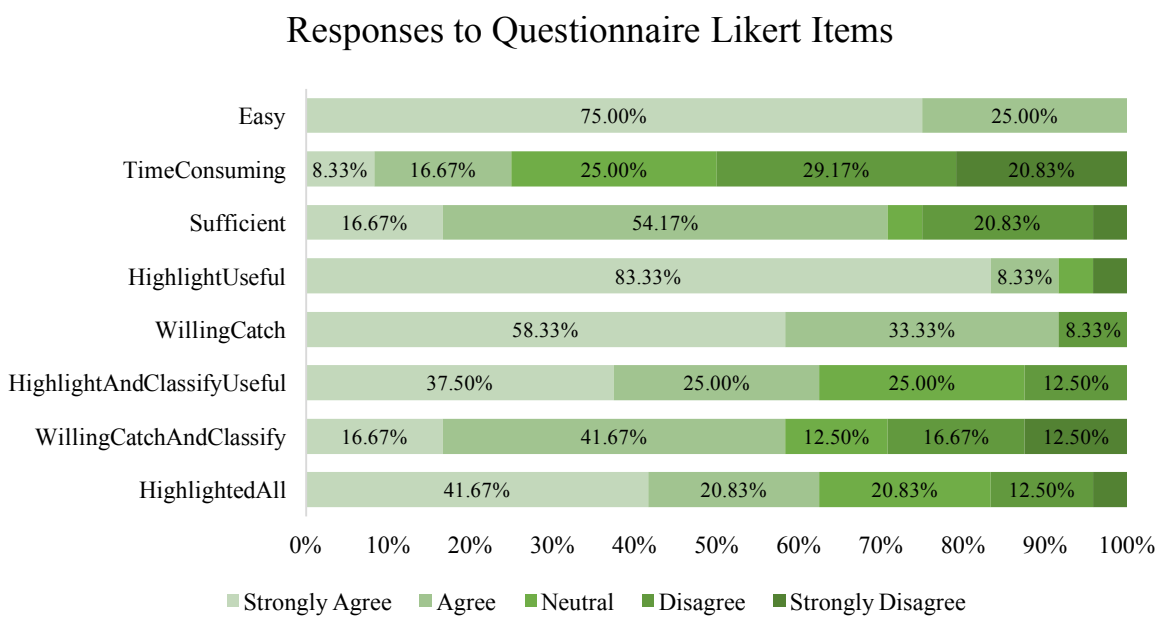


Figure 4.4: Subjective opinions on the process of highlighting and classification in peer review.

Qualitative responses from the open-ended component of the questionnaire gave more insight on how participants felt about the task and interface. A number of participants (6) commented that marking errors was distracting or took more time than just reviewing; however, 2 of those participants expressed positive opinions, claiming that the added time was insignificant and would get faster with experience, and that *"It is time consuming but highly neglected. There should be more done in that aspect."* Despite only being asked to point out low-level errors, participants left comments stating that highlighting was insufficient for pointing out high-level errors such as structure or clarity (6); a few (3) mentioned that they wanted the ability to leave notes on the document. Others commented that highlighting was insufficient for pointing out even low-level errors (2), indicating difficulties with highlighting space or errors of omission. Two participants wanted the ability to classify as well as highlight comments during the review task, while another commented, *"doing so during the reading might be too distracting or annoying."* One participant commented that our highlighting mechanism *"makes life easy."*

4.3 Discussion

We can draw a number of conclusions from our study, despite it being only a preliminary investigation of the tools and techniques we are developing.

4.3.1 Reviewer performance

Our results show that, after a short error-classification task, we can expect a single reviewer to find about 42% of all low-level errors in a paper while performing their regular review, and that three reviewers can be expected to collectively find roughly 80% of all errors. If the highlighting mechanism were incorporated into peer review web software, along with a

classification task prior to review (which may only be necessary once, or perhaps periodically as a "refresher"), our results suggest that the number of errors that make it through to publication could be significantly reduced. Given that frequency of errors did not have a significant effect on errors caught, we expect that this performance rate will generally hold regardless of the number of errors in a paper. However, more tests are needed to determine whether lower error frequencies (e.g., less than 10 errors) would continue to have no effect.

The large discrepancy between primed and unprimed performance (42% versus 15%) indicates that some form of training will be necessary when using this system in peer review. This could take the form of labeling new reviewers as "apprentices" and requiring them to perform a quick classification task before reviewing the paper, or perhaps by periodically asking all reviewers to perform the task. We imagine that with experience, it will not be necessary to perform training for each review.

An interesting result from our questionnaire is that about 40% of participants said they did not highlight every error they saw, perhaps due to repetitiveness or perceiving less benefit for each error they caught. This means that the current single-reviewer rate of 42% could actually be an underestimate of the true proportion, and that a set of three (primed) reviewers could be expected to find even more than 80% of all writing errors.

4.3.2 Overhead of the error noticing task

As we suspected, giving more work to reviewers (by asking them to classify as well as mark errors) resulted in less willingness to perform the activity. However, it is worth noting that over half of the participants said they were still willing to classify errors in real peer review

sessions. If a standard review includes three reviewers, this suggests that it is likely that at least one of them would be willing to provide extra information about the types of errors that were found.

Despite the fact that we designed the highlighting interaction to be minimalist in order to reduce the necessary effort for reviewers to mark errors, our questionnaire results revealed that some reviewers wanted the ability to give even more feedback to authors. This is in line with Nobarany et al.'s findings, which showed that some reviewers are more receptive to helping with writing quality [65]. Our participants wanted the ability to leave comments to clarify their highlights or give higher-level structural advice, and in some cases, they wanted to classify their highlighted errors using our taxonomy. When incorporating annotation into reviewing interfaces, it will be important to support different levels of granularity and expressiveness in annotation, while at the same time supporting minimalist interactions for busy or less committed reviewers.

Although we notice errors naturally while reading, reviewing with the expectation of error noticing, and the act of highlighting errors, is not without some cognitive cost: participants reported that it was distracting at times and that it did take more time than reviewing without paying attention to errors. However, despite these extra costs, the vast majority reported that it was a useful and worthwhile process to include in peer review. By providing simple tools such as a highlighting mechanism to make pointing out errors fast and easy, we may be able to catch more writing errors before they make it through to publication without significantly adding to reviewers' cognitive workload, and perhaps we can convince more reviewers to

provide writing assistance by finding strong evidence that it improves the quality of the scholarly literature.

4.3.3 Error taxonomy

After analyzing how participants classified errors in the priming task, it is clear that our taxonomy needs some work. Participants only agreed with our classifications in about 80% of the cases. In hindsight, it would have been useful to include an additional category for preposition errors, as these make up a large portion of errors made by ESL writers [71]. The most troublesome category was "Sentence Structure," where participants often classified the errors as a miscellaneous issue or a problem with punctuation. We had designed this category as a general catch-all category for poorly-formed sentences, but it is clear we need to "go back to the drawing board" not only with how we differentiate our categories but also how we present them to the participants. For example, we did not explicitly require that participants carefully read the descriptions of each category.

4.3.4 Limitations & future work

The expected single-reviewer primed proportion of 42% was based on the average performance over all of our selected papers. The papers had significantly different writing styles, so we believe the coverage was broad enough to allow us to achieve the 80% benchmark for a fairly large range of papers in terms of length, style, and topic. However, further tests should be conducted to confirm whether this rate varies with longer papers, or with papers from different fields.

It would be interesting to know more about how well authors can interpret highlights and identify their mistakes. Our intention in using the less expressive highlighting method was to

minimize reviewer work, meaning that the onus would be on the author to decipher and fix the errors. We felt this was a fair tradeoff of labour. Even with more difficult errors, we assume that co-authors working together could probably figure out the more ambiguous cases, but further experiments could specifically test how well authors can identify and fix these marked errors. Also of interest is how reviewer highlights in bulk should be presented to the author. For example, should highlights from all reviewers be merged onto the same copy of a document? An interface capable of filtering writing errors by their classification (if any), or by the reviewer who flagged the error, might be useful.

It would be worthwhile to look into how crowdsourcing can help discover low-level errors before publication, for example, by taking the approach of Bernstein et al. [64] and asking Mechanical Turkers to proofread manuscripts that are under submission, with their highlights being returned to the author. Such processes could be automated by web submission interfaces, but careful considerations would need to be taken regarding distributing unpublished work across the web if it may contain scientific errors (because it has yet been reviewed for content) or it is subject to confidentiality restrictions.

Some participants were unsure of how to mark certain errors with highlights, citing errors of omission and spacing errors. We had anticipated that the trickier errors would be marked by highlighting the surrounding context. For example, extra spacing between two paragraphs could be indicated by highlighting the end of the previous paragraph and beginning of the next. We may need to re-think this. One benefit of using less-expressive annotation methods such as highlighting is that it provides less work for the reviewer, and it assumes that the author (perhaps with a bit of work) will be able to decipher what the error is, but at the cost

of potential ambiguities. Future iterations of the reviewing interface might be developed to include equally simple annotation techniques that help to resolve ambiguities.

Participants in our study (graduate students) were relatively inexperienced with formal reviewing for conference or journal papers. It would be interesting to see how well professors and other veteran reviewers perform in comparison.

4.4 Summary

We conducted a study that asked reviewers to mark low-level writing errors they naturally found while reviewing a paper (a phenomenon we have called "error noticing"), while providing them with a simple highlighting mechanism to mark each error. Our results showed that, after a short priming task, a group of three reviewers can be expected to find 80% of all writing errors in a paper, and that reviewers found the act of highlighting writing errors to be highly useful and they would be willing to do it in real peer review sessions. Based on this, we believe that further effort should be made to design lightweight reviewing interfaces for reviewers to mark and possibly even categorize errors they find while reviewing a paper.

Chapter 5: Designing an Interactive Reviewing Tool

The scholarly peer review process currently lacks interactive tools for assisting a reviewer in performing a review of a manuscript. Although reviewing activities such as knowledge acquisition and forming judgements often result in artifacts scattered around (marginal jotted notes, underlines or other annotations), there is currently no tool which can help reviewers organize their collection of thoughts represented as annotations into a report outline, for use when drafting the final (usually plaintext) report.

In Chapter 2, we explored features and requirements for future reviewing tools and concluded that these interfaces should be web-based and will need to work in tandem with Peer Review Management Software (PRMS) to add support for pre-submission anonymity checks, group reviewing activities such as inter-reviewer discussion, and rich annotation and visualization schemes for assisting reviewers in writing their report. In Chapter 3, we described our simple proof-of-concept reviewing tool, and in Chapter 4 we showed that our tool helped reviewers in a controlled laboratory study to catch 80% of all writing errors in a manuscript simply by including a basic highlighting mechanism.

In this chapter, we outline preliminary steps towards extending our earlier prototype to better support the private reviewing activities of a single reviewer. We begin by describing a flexible annotation model implemented using DOM manipulation that will serve as a basis for later peer-review-specific annotations and workflows. Then, we provide a conceptual design for a novel reviewing experience using annotations on a manuscript. The work presented here is preliminary and still in progress and does not yet consider elements of

collaborative reviewing activities nor integration with PRMS systems, but we view it as a first step towards creating interactive reviewing interfaces. We have (unofficially) given the prototype the moniker “Pear Review” for its resemblance to “peer review” and our research lab’s affinity for naming experimental prototypes after types of fruit.

5.1 Highlighting

Pear Review supports basic highlighting of contiguous sections of text in a manuscript. To create a highlight, the user selects text in the browser and presses the spacebar to create the highlight; we found this method to be simple and well liked in our experiment described in Chapter 4. Highlights can be selected by clicking on them (the highlight darkens and text becomes underlined) and deleted by pressing the delete key when a highlight is selected. Pear Review extends basic highlighting in three ways: (a) it colour-codes highlights to the types of issues they represent, (b) it provides support for disambiguating overlapping highlights using visual markers that we call *knobs*, and (c) it uses these knobs to provide spatial awareness of where and which types of issues are occurring on a page.

After creating a highlight, the user can press the tab key to open up a menu to choose from a list of categories in which to classify the issue (Figure 5.1). Continuing to press the tab key cycles through the options and updates the colour of the highlight. To finalize the category selection, the user presses the enter key or clicks elsewhere on the page. Writing errors can be highlighted and automatically classified as such by pressing the tilde (~) key shortcut, although they can still be classified using the standard tab menu.

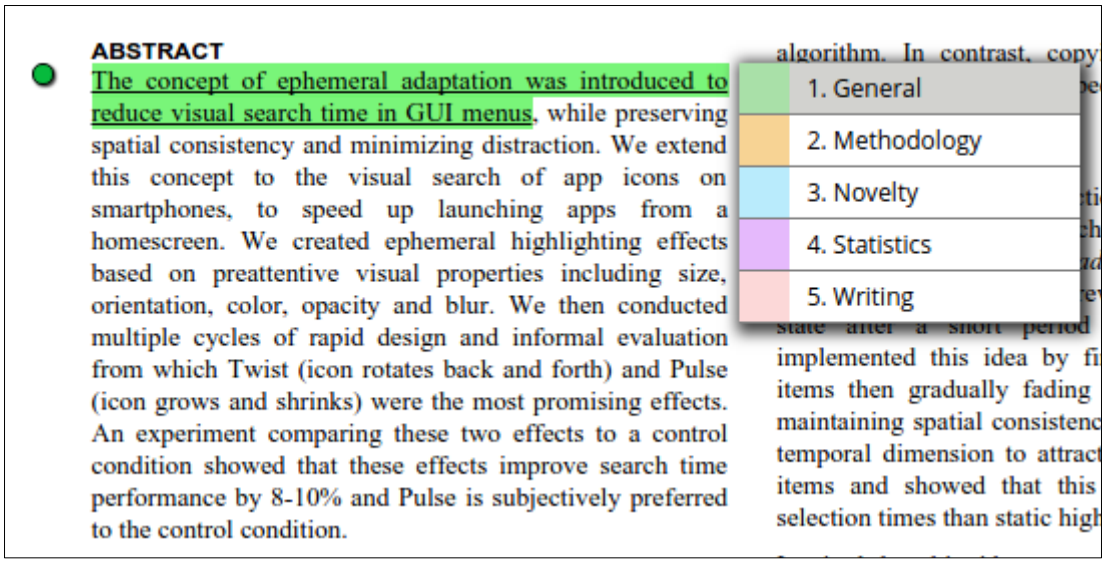


Figure 5.1: A selected highlight can be categorized by clicking the tab button and cycling through a list of colour-coded categories in order to classify an annotation.

With highlighting annotations in general, it can be difficult to disambiguate between multiple overlapping highlights. For example, a reviewer may find and highlight two writing errors in the same sentence and also highlight the entire sentence due to an issue with its content. Pear Review provides a number of ways to disambiguate between such overlaps. All highlights contain a degree of transparency, so it is relatively obvious when one highlight is contained within another because the inner one will appear darker if it is the same colour (different colours overlapping are easier to disambiguate due to colour mixing). However, this transparency technique quickly breaks down when there are many highlights that overlap in different ways. To make the boundaries of different highlights more distinct, Pear Review uses hover effects that slightly darken the colour of the highlight currently under the cursor and underlines the associated anchor text (Figure 5.2).

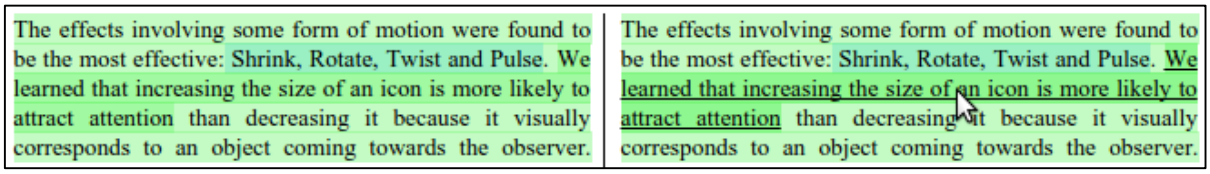


Figure 5.2: Overlapping highlights (left) and a cursor hover effect (right) that darkens the highlight's colour and underlines the text to help disambiguate annotation boundaries.

Finally, Pear Review places small circular nodes (called *knobs*) in the margin where a highlight's text selection begins. These knobs are the same colour as the highlight and show how many highlights (issues) exist on that line. Hovering over the knob greys out all other highlights so the user can focus solely on the hovered annotation (Figure 5.3).

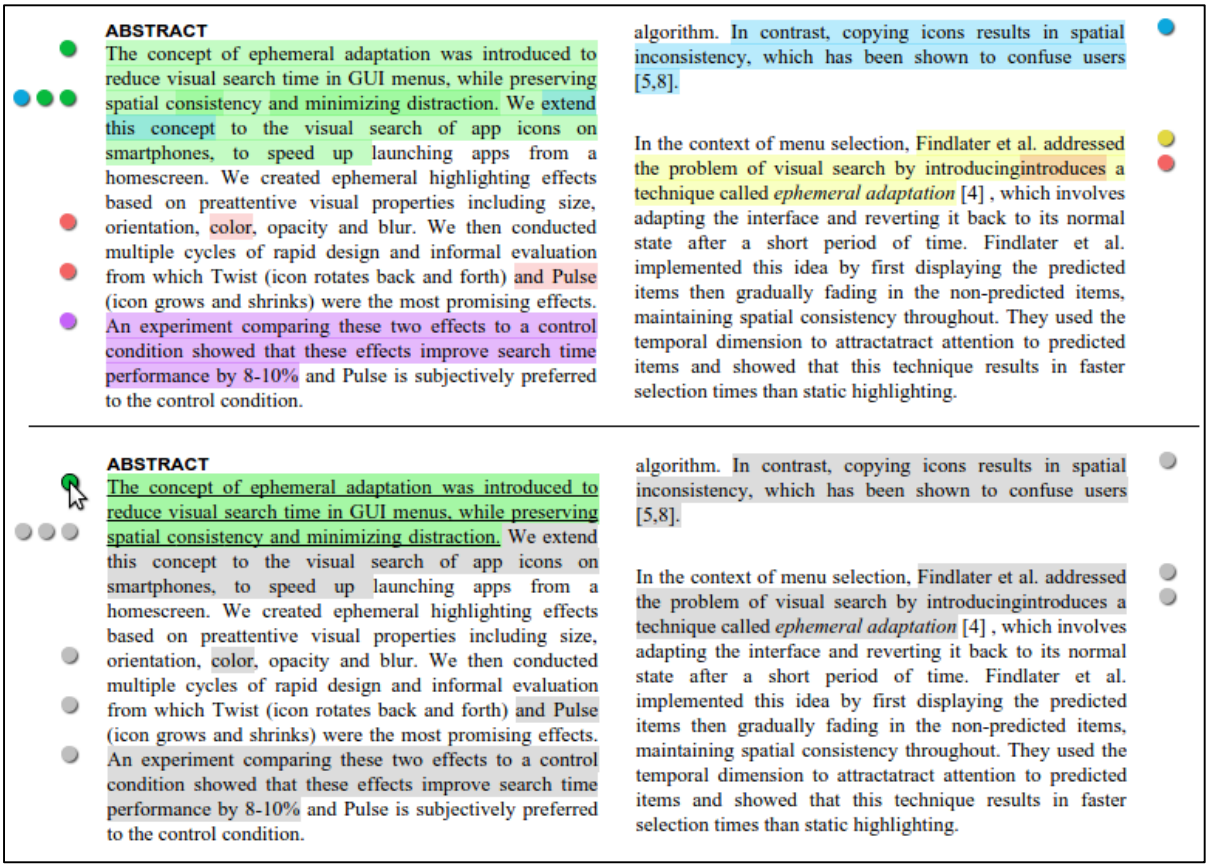


Figure 5.3:"Knobs" on the margins of the manuscript help disambiguate overlapping annotations and provide the user with a high-level view of the different types and frequencies of issues occurring on a page. Hovering over an annotation's knob will grey out any other annotations.

5.2 Commenting

Creating comments in Pear Review is similar to creating highlights because comments are simply an extension of highlights. To create a comment, the user selects a contiguous section of text and presses the enter key (rather than the spacebar for a highlight). This creates a highlight over the selected text and opens up a comment box right below it (Figure 5.4). The user can type a comment then close the comment box by pressing the escape key or clicking anywhere else on the manuscript. A highlight can be converted into a comment by first selecting the highlight then pressing the enter key. Comment boxes reappear whenever their highlight is selected and are hidden when the selection is lost. The marginal knobs for highlights differentiated from comment knobs; highlight knobs are circular and comment

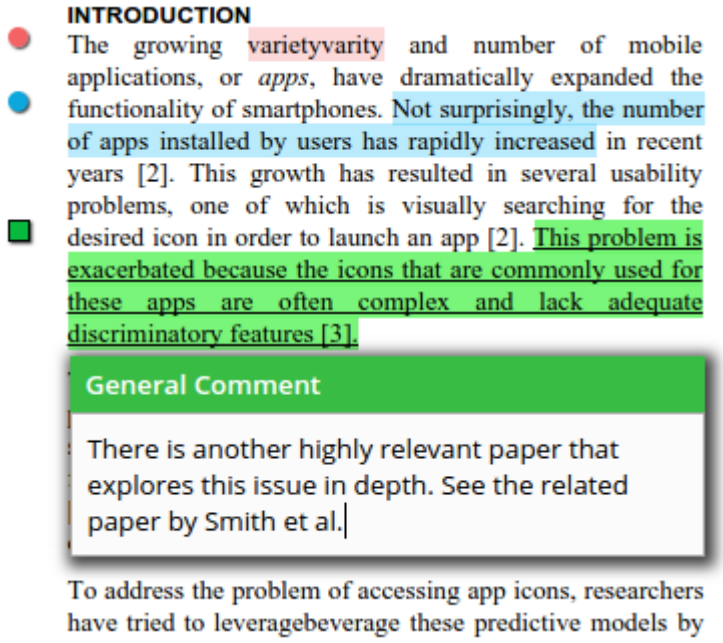


Figure 5.4: A comment annotation in Pear Review, created by selecting text and clicking Enter or pressing Enter on a highlight annotation. Knobs for comments are square. Clicking outside the comment will minimize it.

knobs are square (resembling he appearance of a sticky note).

5.3 Interactive Report Writing with Annotation Visualizations

A key feature in Pear Review will be enabling the reviewer to use annotations made on the manuscript to help the reviewer consolidate and organize their thoughts and judgements about a manuscript to facilitate writing the summarized report. We present an initial conceptual design (Figure 5.5) for organizing annotations made on a manuscript into a form of outline, and using this outline to help a reviewer write a plaintext report (assuming that integration with a PRMS is not yet available and the submission requires a plaintext).

First, a reviewer conducts a reviewing session on a manuscript, using annotations to record their thoughts about the manuscript as they read. After reading and annotating, the user transitions into a view where they can see each annotation they have made, but in a representation outside of the manuscript context. The user can easily jump between any

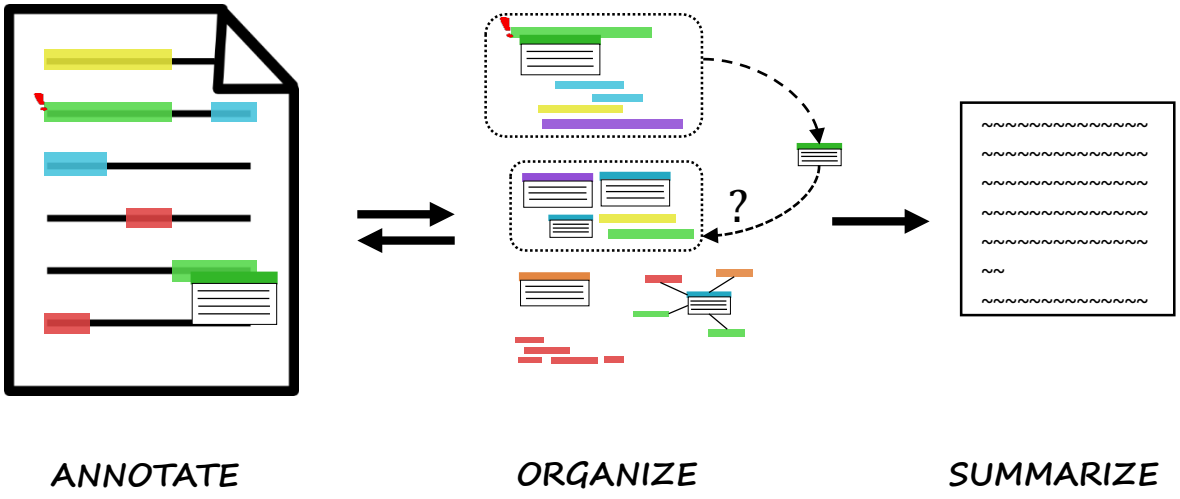


Figure 5.5: Conceptual design for the interactive report-writing feature in Pear Review. After annotating a paper, a reviewer can directly manipulate the annotations into an outline to help them form their written report.

annotation's representation in this view and its anchored location in the paper by clicking on it. Hovering over an annotation representation in this view shows a preview pane of the annotation's anchor within the manuscript along with its surrounding context.

The annotations in this organization view are filterable by type of issue, importance, type of annotation, ordinal location in the paper, or other annotation metadata. The user has the freedom to directly manipulate the annotations by organizing them into a hierarchical outline, affinity diagram, or mind map. The main purpose of this organization view is to help the reviewer visualize their annotations from a higher level, helping them to collect and organize their main points to help them draft their summarized report. The reviewer can switch back and forth between the manuscript and the annotation organization view at any time.

The area for writing the plaintext report is also displayed in the annotation organization view so the reviewer does not need to go back and forth between the organization view and their draft. Because comments created on the manuscript may contain well-written points that are transferrable into the report, comments are directly insertable from the comment annotations into the report draft, where they may then be further revised by the reviewer to match the style of the surrounding writing.

The user is able to append a list of the writing errors that they discovered and flagged in the manuscript (as described earlier) to the end of the report; the system automatically generates textual descriptions the errors, the page and paragraph in which they occur, and possibly the type of error if the reviewer felt like categorizing the error into a more granular types (e.g. misspelling, formatting, malapropism, etc.).

Because some PRMS systems use multiple textboxes for different parts of a report, the reviewer is able to customize the area containing the report draft to have multiple text entry areas or checked options that correspond to their conference or journal's report style (if not already found in the existing library of conference and journal report templates). Once the reviewer finishes writing a report, they copy and paste the text from the report area into the report submission page of their PRMS (or submit it directly in the interface assuming integration with a PRMS system).

5.4 Future Work

Pear Review is still in its early design phases and there is much development and design left to do. A final version should include filtering capabilities on the manuscript view, for example, to filter out low-level writing errors or other annotations the reviewer does not wish to see while reviewing higher-level issues. More review-specific annotations should be implemented such as those described in Chapter 2, for example, an annotation that indicates missing relevant work, or more general annotations such as tacit marks or drawings. Annotations should be more functional and should possibly support multiple text anchors per annotation, anchoring to figures or tables, and meta-data indicating the relative importance of some annotations over others. The semantic categories of manuscript issues and their colours should be customizable. Special attention should be paid to the visual aesthetics and user experience, and default colours should be used appropriately to avoid distraction; there should be additional support for colour-blind reviewers. Most importantly, a full prototype should be validated through usability studies and tested to see whether the novel annotation

organization view can improve reviewers' efficiency and whether our interface provides a better user experience for reviewing and drafting a report.

Bibliography

[1] M. Hojat, J. S. Gonnella, and A. S. Caelleigh, “Impartial judgment by the ‘gatekeepers’ of science: fallibility and accountability in the peer review process,” *Adv. Health Sci. Educ. Theory Pract.*, vol. 8, no. 1, pp. 75–96, 2003.

[2] J. M. Campanario, “Peer Review for Journals as it Stands Today--Part 1,” *Sci. Commun.*, vol. 19, no. 3, pp. 181–211, 1998.

[3] J. M. Campanario, “Peer Review for Journals as it Stands Today--Part 2,” *Sci. Commun.*, vol. 19, no. 4, pp. 277–306, 1998.

[4] L. Bornmann, “Scientific Peer Review,” *Annu. Rev. Inf. Sci. Technol.*, vol. 45, no. 1, pp. 197–245, 2011.

[5] R. Spier, “The history of the peer-review process.,” *Trends Biotechnol.*, vol. 20, no. 8, pp. 357–358, 2002.

[6] D. a Kronick, “Peer review in 18th-century scientific journalism.,” *JAMA*, vol. 263, no. 10, pp. 1321–1322, 1990.

[7] J. C. Burnham, “The evolution of editorial peer review,” *JAMA J. Am. Med. Assoc.*, vol. 263, no. 10, pp. 1323–9, 1990.

[8] D. Kennefick, “Einstein versus the Physical Review,” *Phys. Today*, vol. 58, no. 9, pp. 43–48, 2005.

[9] M. Nielsen, “Three myths about scientific peer review,” 2009. [Online]. Available: http://michaelnielsen.org/blog/three-myths-about-scientific-peer-review/. [Accessed: 12-Dec-2015].

[10] M. D. McIlroy, “A Research UNIX Reader: Annotated Excerpts from the Programmer’s Manual, 1971-1986,” *Pic.Plover.Com*, pp. 1971–1986, 1986.

[11] “Open Journal Systems.” [Online]. Available: https://pkp.sfu.ca/ojs/. [Accessed: 01-Jul-2016].

[12] “OpenConf.” [Online]. Available: https://www.openconf.com/. [Accessed: 01-Jul-2016].

[13] “Easy Chair.” [Online]. Available: http://www.easychair.org/. [Accessed: 01-Jul-2016].

[14] "Scholastica." [Online]. Available: https://scholasticahq.com/. [Accessed: 01-Jul-2016].

[15] "ExOrdo." [Online]. Available: https://www.exordo.com/. [Accessed: 01-Jul-2016].

[16] P. K. Project, "Open Conference Systems." [Online]. Available: https://pkp.sfu.ca/ocs/.

[17] "alt.chi 2016." [Online]. Available: https://chi2016.acm.org/wp/alt-chi/. [Accessed: 01-Jul-2016].

[18] "Precision Conference Solutions." [Online]. Available: https://precisionconference.com/. [Accessed: 01-Jul-2016].

[19] S. Nobarany and K. S. Booth, "Use of Politeness Strategies in Signed Open Peer Review," *J. Assoc. Inf. Sci. Technol.*, vol. 66, no. 5, pp. 1048–1064, 2015.

[20] "publons." [Online]. Available: https://publons.com/. [Accessed: 01-Jul-2016].

[21] V. Bush, "As we may think," *SIGPC Note.*, vol. 1, no. 4, pp. 36–44, 1979.

[22] E. F. Churchill, J. Trevor, S. Bly, L. Nelson, and D. Cubranic, "Anchored Conversations: Chatting in the Context of a Document," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2000, pp. 454–461.

[23] M. J. Adler, "How to mark a book," *Saturday Review of Literature 6*, pp. 250–252, 1940.

[24] J. L. Wolfe and C. M. Neuwirth, "From the Margins to the Center: The Future of Annotation," *J. Bus. Tech. Commun.*, vol. 15, no. 3, pp. 333–371, 2001.

[25] I. A. Ovsiannikov, M. A. Arbib, and T. H. McNeill, "Annotation technology," *Int. J. Hum. Comput. Stud.*, vol. 50, no. 4, pp. 329–362, 1999.

[26] C. C. Marshall, "Annotation: from paper books to the digital library," in *Proceedings of the second ACM international conference on Digital libraries - DL '97*, 1997, pp. 131–140.

[27] C. Neuwirth, D. Kaufer, R. Chimera, and T. Gillespie, "The Notes Program: A Hypertext Application for Writing from Source Texts," in *Proc. Hypertext '87, November 13-15, 1987, Chapel Hill, North Carolina, USA*, 1987, pp. 121–141.

[28] K. O'Hara and A. Sellen, "A Comparison of Reading Paper and On-Line Documents," in *Proceedings of CHI'97, the ACM SIGCHI Conference on Human Factors in Computing Systems*, 1997, pp. 335–342.

[29] J. L. Wolfe, “Effects of Annotations on Student Readers and Writers,” in *Proceedings of the fifth ACM conference on Digital libraries - DL ’00*, 2000, pp. 19–26.

[30] B. N. Schilit, G. Golovchimlq, and M. N. Price, “Beyond paper: Supporting active reading with free form digital ink annotations,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI ’98)*, 1998, pp. 249–256.

[31] R. Kopak and C.-N. Chiang, “Annotating and linking in the Open Journal Systems,” *First Monday*, vol. 12, no. 10, 2007.

[32] C. M. Neuwirth, D. S. Kaufer, R. Chandhok, and J. H. Morris, “Issues in the design of computer support for co-authoring and commenting,” in *Proceedings of the 1990 ACM conference on Computersupported cooperative work*, 1990, pp. 183–195.

[33] J. J. Cadiz, A. Gupta, and J. Grudin, “Using Web annotations for asynchronous collaboration around documents,” in *Proceedings of the 2000 ACM conference on Computer supported cooperative work CSCW ’00*, 2000, pp. 309–318.

[34] C. M. S. Weng, D. W. McDonald, and J. H. Gennari, “A Collaborative Clinical Trial Protocol Writing System,” in *Proceedings of MedInfo’2004*, 2004.

[35] Q. Zheng, K. Booth, and J. McGrenere, “Co-Authoring with Structured Annotations,” in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, 2006, pp. 131–140.

[36] I. Glover, Z. Xu, and G. Hardaker, “Online annotation - Research and practices,” *Comput. Educ.*, vol. 49, no. 4, pp. 1308–1320, 2007.

[37] C. Weng and J. H. Gennari, “Asynchronous Collaborative Writing through Annotations,” in *Proceedings of the 2004 ACM conference on Computer supported cooperative work - CSCW ’04*, 2004, pp. 578–581.

[38] T. A. Phelps and R. Wilensky, “Multivalent Annotations,” in *Research and Advanced Technology for Digital Libraries*, 1997, pp. 287–303.

[39] C. C. Marshall, “Toward an ecology of hypertext annotation,” in *Proceedings of the ninth ACM conference on Hypertext and hypermedia HYPERTEXT ’98*, 1998, pp. 40–49.

[40] R. Furuta and E. Urbina, “On the Characteristics of Scholarly Annotations,” in *Proceedings of the thirteenth ACM conference on Hypertext and hypermedia*, 2002, pp. 78–79.

[41] C. C. Marshall and A. J. B. Brush, “From personal to shared annotations,” in *CHI ’02 extended abstracts on Human factors in computing systems*, 2002, pp. 812–813.

[42] J. Laurens, “Direct and reverse synchronization with SyncTEX,” *TUGBoat*, vol. 29, pp. 365–371, 2008.

[43] “ShareLaTeX.” [Online]. Available: https://www.sharelatex.com/. [Accessed: 01-Jul-2016].

[44] “Xodo.” [Online]. Available: https://www.xodo.com/. [Accessed: 16-Jul-2016].

[45] “Xodo (Slack integration beta).” [Online]. Available: https://app.xodo.com/. [Accessed: 16-Jul-2016].

[46] S. Iga and M. Shinnishi, “SnapShoot: Integrating semantic analysis and visualization techniques for web-based note taking system,” *Conf. Res. Pract. Inf. Technol. Ser.*, vol. 60, pp. 161–167, 2006.

[47] “Google Docs.” [Online]. Available: https://www.google.com/docs/about/. [Accessed: 01-Jul-2016].

[48] “pdfcomment.” [Online]. Available: https://www.ctan.org/pkg/pdfcomment. [Accessed: 01-Jul-2016].

[49] “todonotes.” [Online]. Available: https://github.com/henrikmidtiby/todonotes. [Accessed: 01-Jul-2016].

[50] “TrackChanges.” [Online]. Available: http://trackchanges.sourceforge.net/.

[51] “PleaseReview.” [Online]. Available: http://www.pleasetech.com/.

[52] “PDF Annotator.” [Online]. Available: https://www.pdfannotator.com/. [Accessed: 01-Jul-2016].

[53] “Foxit Reader.” [Online]. Available: https://www.foxitsoftware.com/products/pdf-reader/. [Accessed: 01-Jul-2016].

[54] J. R. Wright and K. Leyton-brown, “Mechanical TA: Partially automated high-stakes peer grading,” *Sigcse ’15*, pp. 96–101, 2015.

[55] “PDF.js.” [Online]. Available: https://mozilla.github.io/pdf.js/. [Accessed: 01-Jul-2016].

[56] “WebViewer.” [Online]. Available: https://www.pdftron.com/webviewer/. [Accessed: 16-Jul-2016].

[57] “IEEE VGTC Conference Proceedings.” [Online]. Available: http://junctionpublishing.org/vgtc/. [Accessed: 09-Aug-2016].

[58] A. M. Tremonti, “The Current: Good writing in the 21st century needs clarity, says Steven Pinker,” CBC Radio.

[59] J. J. Pollock and a Zamora, “Collection and Characterization of Spelling Errors in Scientific and Scholarly Text,” *J. Am. Soc. Inf. Sci.*, vol. 34, no. 1, pp. 51–58, 1983.

[60] J. H. Sweetland, “Errors in Bibliographic Citations: A Continuing Problem,” *Libr. Q. Information, Community, Policy*, vol. 59, no. 4, pp. 291–304, 1989.

[61] M. Scott and J. Turner, “Problematising Proofreading,” *Zeitschrift Schreiben*, 2008. [Online]. Available: http://www.zeitschrift-schreiben.eu/Beitraege/scott_Proofreading.pdf.

[62] J. D. Gould and N. Grischkowsky, “Doing the same work with hard copy and with Cathode-Ray Tube CRT computer terminals,” *Hum. Factors J. Hum. Factors Ergon. Soc.*, vol. 26, no. 3, pp. 323–337, 1984.

[63] J.-F. Rouet, *Hypertext and Cognition*. Psychology Press, 1996.

[64] M. S. Bernstein, G. Little, R. C. Miller, B. Hartmann, M. S. Ackerman, D. R. Karger, D. Crowell, and K. Panovich, “Soylent: A Word Processor with a Crowd Inside,” in *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, 2010, pp. 313–322.

[65] S. Nobarany, K. S. Booth, and G. Hsieh, “What Motivates People to Review Articles? The Case of the Human-Computer Interaction Community,” *J. Assoc. Inf. Sci. Technol.*, 2015.

[66] J. G. Ruiz, C. Candler, and T. A. Teasdale, “Peer Reviewing E-Learning : Opportunities , Challenges , and Solutions,” vol. 82, no. 5, pp. 503–507, 2007.

[67] S. Nobarany, “Policies, Practices, and Potentials For Computer Supported Scholarly Peer Review,” 2015.

[68] J. Nielsen and T. K. Landauer, “A mathematical model of the finding of usability problems,” in *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI ’93*, 1993, pp. 206–213.

[69] W. H. Bryant, “Typical Errors in English Made by Japanese ESL Students,” *Japan Assoc. Lang. Teach. J.*, vol. 6, no. 1, pp. 1–18, 1984.

[70] A. Y. W. Chan, “Toward a Taxonomy of Written Errors: Investigation Into the Written Errors of Hong Kong Cantonese ESL Learners,” *TESOL Q.*, vol. 44, no. 2, pp. 295–319, 2010.

[71] A. Rozovskaya and D. Roth, “Annotating ESL errors: Challenges and rewards,” *Proc. NAACL HLT 2010 fifth …*, no. June, pp. 28–36, 2010.

[72] C. James, *Errors in Language Learning and Use: Exploring Error Analysis*. Routledge, 2013.

[73] A. J. B. Brush, D. Bargeron, A. Gupta, and J. J. Cadiz, “Robust annotation positioning in digital documents,” in *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI ’01*, 2001, pp. 285–292.

[74] S. Bateman, R. Farzan, P. Brusilovsky, and G. McCalla, “OATS: The Open Annotation and Tagging System,” in *Proceedings of the Third Annual International Scientific Conference of the Learning Object Repository Research Network*, 2006.

[75] “W3C Web Annotation Working Group.” [Online]. Available: https://www.w3.org/annotation/. [Accessed: 06-Jul-2016].

Appendix A: Anonymization Tools for LaTeX

As a proof of concept, we implemented two simple scripts in Python that wrap the LaTeX chain of commands to help authors anonymize their manuscripts before submission.

A.1 AnonBib

AnonBib anonymizes bibliographic entries in a manuscript that need to be obscured to satisfy anonymity constrains. For each BibTeX entry ID in a supplied ".anon" file, AnonBib replaces each matching entry in the user's BibTeX file with an anonymous placeholder (Figure A.1). Authors would presumably maintain their own .anon file (as they typically do with their .bib file) and keep it up to date with own published manuscripts. They would then run AnonBib before generating the "camera ready" version of their manuscript. Although not currently supported, a simple improvement would be accepting multiple .anon files so that coauthors could anonymize each of their own citations in one pass.

AnonBib could be used more transparently if integrated into a cloud coauthoring system like ShareLaTeX. ShareLaTeX co-authors could each have their own configured .anon file and compiling the manuscript could automatically use both files to anonymize the bibliography.

REFERENCES

1. Anony, M. Journal of anonymity.
2. Anony, M. Anonymous title. In *Journal of Anonymity*, ANON (2016), 120–136.
3. Bowman, D. A., Wingrave, C. A., Campbell, J., Ly, V., and Rhoton, C. Novel uses of pinch gloves for virtual environment interaction techniques. *Virtual Reality 6*, 3 (2002), 122–129.

Figure A.1: References section of a manuscript anonymized by AnonBib. Entries in a separate .anon file determine which BibTeX entries are replaced with generic placeholders.

A.2 AnonTeX

AnonTeX is script that wraps the LaTeX chain of commands to produce a PDF manuscript with an anonymous author blocks. It parses the supplied LaTeX file and replaces each character of text within the author blocks with a configurable placeholder character such as an asterisk (Figure A.2). The repeating character fills up each line of the author block and fills spaces so that readers cannot deduce the author's name or institution by the length of words and spaces. We went with the approach of replacing existing characters in the LaTeX file rather than arbitrary blocks of text that take up space in order to preserve the original look and feel of the paper and avoid unintended layout changes.

Our approach has some limitations: it does not obscure the number of authors on a paper and is not fully robust against non-standard author blocks in LaTeX markup. Future work should explore how we can both preserve the spatial layout of the paper while totally obscuring all author information, including the number of authors.

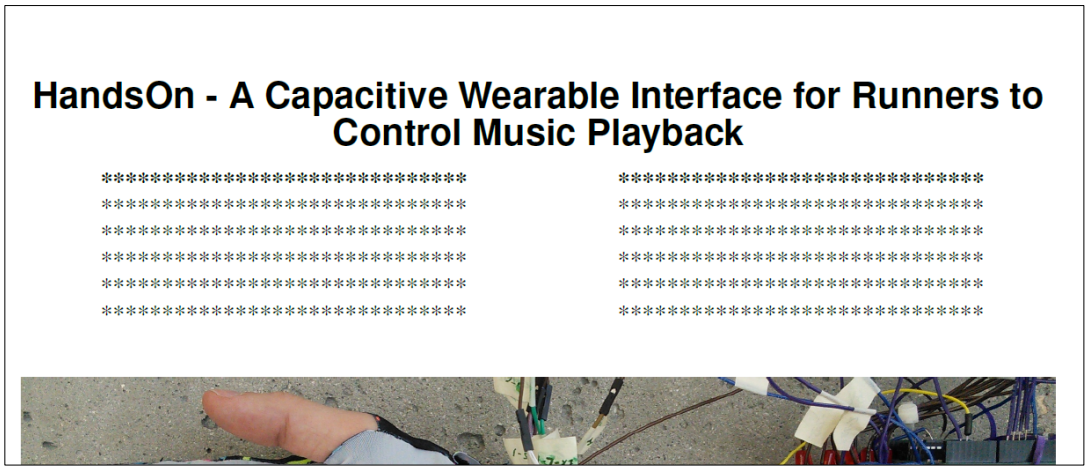


Figure A.2: Anonymized author blocks generated by AnonTeX. The author names and institutions are replaced with placeholder characters. The original number of lines in each block along with the number of authors is preserved, but the content is obscured.

Appendix B: Highlighting the DOM

B.1 HTML Structure

This is a highlight within a single paragraph element's text node.

HTML

```
<p>This is a <mark data-hl-id="1">highlight within a single paragraph
element's</mark> text node.</p>
```

This highlight spans two paragraphs elements.

It is represented by two marks with the same id.

HTML

```
<p>This highlight spans two <mark data-hl-id="2">paragraph elements.</mark></p>
<p><mark data-hl-id="2">It is represented by two marks</mark> with the same
id.</p>
```

B.2 Simple JavaScript Implementation

The following is a JavaScript snippet that creates a highlight over a continuous text selection.

JavaScript + jQuery

```
function highlight() {
    var selection = window.getSelection().getRangeAt(0);
    var nodes = getTextNodesInRange(selection);

    nodes.forEach(function(node, i) {
        var range = document.createRange();
        range.setStart(node, i === 0 ? selection.startOffset : 0);
        range.setEnd(node, i === nodes.length – 1 ? selection.endOffset
            : node.length);
        range.surroundContents($("<mark>", {"data-hl-id": nextId++})[0]);
    });
}
```

The `highlight()` method works by fetching all text nodes belonging to elements in the current browser text selection and creating possibly multiple mark elements to represent a single highlight, usually in response to some key press by the user. The algorithm by its nature already has support overlapping highlights; creating a highlight over another will simply nest mark tags so that they both wrap the same text node. However, the overlaps cannot be visualized without additional styling considerations, for example, using transparent background colours or borders for mark elements (we explored this in Section 5.1).

Certain considerations need to be made for more complex highlighting behaviors (the snippet above is the most basic example). For instance, hiding and showing highlights requires recalculating the start and end offsets of a highlight because the offsets are based on the raw HTML content, and adding new highlights before existing ones can alter the offsets at which the highlight should reappear. Furthermore, when deleting highlights it becomes necessary to renormalize the text nodes' parent elements (DOM elements have a `normalize()` method) to reattach the fractured text nodes caused by deleting the marks.

Appendix C: Supplementary Materials and Analysis

C.1 Simplified Review Form

The review form consists of the following Likert questions with the scale: *Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree*.

1. The paper was interesting and engaging.
2. The paper is well organized, which contributed to my understanding of the content.
3. The abstract provided a concise and sufficient summary of the work.
4. The writing is clear, well-formatted, and high quality.
5. I have enough background knowledge to understand the content from a high level.
6. I have enough background knowledge to understand the low-level and technical details in the paper.
7. The paper gives sufficient background information and relevant work.
8. The paper has a strong and clear methodology.
9. The arguments are cogent, and conclusions made are supported by evidence or data.
10. The paper makes clear conclusions and described avenues for future work.
11. I am convinced that this work is relevant and impactful.
12. In my opinion, I feel this paper is suitable for a course on peer review.

There was a single open question at the end labelled: “General comments (if any).”

C.2 Sample Paper with Low Error Frequency

More Human than Human? A Visual Processing Approach to Exploring Believability of Android Faces

Anonymous Author 1
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 2
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 3
Institution of Anon.
Anon. Address
email@anon.ano

ABSTRACT

The issue of belivability is core to android science, the challenge of creating a robot that can pass as a near human. While researchers are making great strides in improving the quality of androids and their likeness to people, it is simultaneously important to develop theoretical foundations behind believability, and experimental methods for exploring believability. In this paper, we explore a visual processing approach to investigating the believability of android faces, and present results from a study comparing current-generation android faces to humans. We show how android faces are still not quite as believable as humans, and provide some mechanisms that may be used to investigate and compare believability in future projects.

INTRODUCTION

Androids are a class of robots that have the ultimate goal of being able to approximately pass for human [16]. To accomplish this, androids will eventually have to look, move, and interact the same as everyday people. When these goals aren't met, not only are androids easily identifiable as non-human, but interaction can suffer in other ways, such as the android appearing eerie or making people uncomfortable (often referred to the Uncanny Valley problem [15]). Moving forward, android developers will need a solid understanding of which features and characteristics impact believability of their androids, understanding of the underlying perception mechanisms that impact believability, and tools and methods to help diagnose and determine their own android's believability.

In this paper, we take initial step toward this goal by exploring visual perception and processing of android faces. We purposely select a heavy simplification of the broader problem, focusing on the visual perception of static images of android faces. This serves as an initial base case where believability is arguably more easily achievable than with real robots, motion, and interaction.

We present a visual processing discussion and initial foundations explaining how people may process android faces, and conducted a study based on this theory comparing human faces to faces of current-generation androids. While androids are becoming impressively believable, our results show that – as predicted by our theory – people are still faster at identifying human faces, find android faces more eerie, and make more mistakes with android faces.

This work provides a initial step toward building a theoretical foundation for the believability of android faces. At the very least, we have shown how simple studies examining face-identification times and error rates can be used to test android faces and infer potential believability.

RELATED WORK

The general study of how robotic design impacts interaction is well established in field, e.g., comparing zoomorphic and anthropomorphic designs in terms of perceived animacy [1], or building frameworks for appropriate and believable social robot behaviors (e.g., [17]). We propose to extend this direction by specifically addressing the believability of android faces.

Most work on the believability of robots surrounds the eeriness problem (often called the uncanny valley problem). Since first postulated [22], this issue has been contentious [13], and many researchers have looked to unpack the issue in terms of robot dimensions, e.g., morphology [2, 8, 15], or realism and iconicity [5, 6]. While eeriness is inevitably a part of believability of android faces, we take a more wholistic approach where eeriness is but one part of the issue.

There has been limited work in robotics and animation that looks at how people visually process artificial faces. One work looked at how people process a real face, an animated face, and various points "morphed" in between [8], and found that people took longer to classify ambiguous faces than clearly human or animated, with classification time decreasing as the ambiguity decreased. In our work we aim to continue this direction and explicitly target androids.

Some researchers have looked at the impact of faces, for example how modifying facial features can impact response such as perceived attractiveness of an agent [9], and similar work looked at how people can apply gender stereotypes to a robot based solely on the haircut [12]. We extend this direction by addressing believability.

1

VISUAL PROCESSING

Humans are hard wired to see and find faces, even where none exist: this can be illustrated by a common face-finding exercise in doodles [19], and has been linked to a human need for social interaction [10] – people can recognize a face in about 350ms [24]. As such, one would expect the creation of believable android faces to be fairly simple. Unfortunately, although people can easily understand even a crude android design as having a face, believability is a separate problem.

When a person sees a human face – or a face they believe may be human, such as an android – they conduct face recognition processing, to determine if they know the face. Unlike the simple task of finding faces, this requires a great deal more sophistication given the range of differences and subtleties between people's faces. There is a body of work examining this processing, much of it dealing with how people scan and fixate on a potential face (e.g., [3, 24]).

When a person detects a face which at first-glance appears to be human, but deeper face-recognition processing detects a problem, we have an "expectancy violation" (21). Such violations in processing draws a person's attention (even at the subconscious level) to the violation as a means of investigating why the violations happened; as such, we can expect that faces which are not quite normal will take more time for a person to process, given this violation. Further, expectancy violations have also been shown to impact anthropomorphism and believability in other contexts [21], and so we can anticipate similar results here. Arguably, if this violation is jarring and the drop in anthropomorphism is large, this may contribute to the eeriness problem.

There are standard ways that people scan faces, for example, many people initially look below the eyes [24], and usually look at the nose, mouth, and cheeks in some order [3], commonly forming a distinct "T" pattern (eyes and then down) [23]. There is also evidences that problems with eyes are more salient than other features [14]. Individual differences (e.g., based on culture [18] or gender [4]) do exist, which is important to consider for studying believability of androids; for example, some studies show women as having superior face processing [4] so they may be more difficult to deceive with androids. People also have varying tendencies to anthropomorphize (dispositional anthropomorphism) – to give non-human things such as images and potential faces human qualities [11]. Despite these differences, it may still be feasible to study general eye-gaze patterns to help diagnose why an android face is not seen as human. For example by detecting uncharacteristically long fixations or fixation order across many people. In addition, it may be useful to measure a person's disposition to anthropomorphize as an important source of error in data analysis, where people with lower disposition may perhaps be better at detecting issues with an android face.

Some research has purposely distorted human faces to study results and infer about visual processing. For example, by inverting faces (upside-down) or components (inverting the mouth or ryes only), to separate whole-face from component processing [7]. Such techniques, including as hiding the eyes or mouth respectively and doing recognition tests (e.g., as in [7, 14, 25]), can be useful to diagnose components of an android face.

EXPLORATORY PILOT STUDY

We conducted an initial study looking at people's processing of android and human faces. As a pilot, we focused simply on people's classification of static images of faces as either android or human, following the experiment design of [8], and conducted a series of exploratory analyses.

As a primary base case, we wanted to investigate if currentgeneration android faces are sufficiently believable as human. Also, based on our visual processing background given above we hypothesized that people would take longer to process and classify android faces, would make more mistakes (higher error rate), and, due to the increased ambiguity, would find the android faces more eerie. We anticipated that a person's disposition toward anthropomorphism (general tendency to anthropomorphize) would negatively correlate with response time, as we postulated that they would more readily accept the android face as human, and would have higher error rates with android faces given their potential tendency to mistake them for human. Further, we expected that female participants would have lower error rates and quicker response times due to potential face recognition advantages.

Methodology and Procedure

We recruited fourteen participants from our general university population aged 18-58 (Mdn=20.5), with an equal male / female split, and paid them $10 for their participation. Participant nationalities were primarily Canadian, and also included Nigerian, Chinese, Brazilian, and Pakistani participants.

Experiments were conducted with one participant at a time. Participants were briefed about the study and we obtained informed consent. They then sat at a desk at a fixed distance (15cm) from a 24" wide-screen 16:10 monitor and fixed location (centered); we used a chin-rest (sanitized between participants) to ensure this, and participants wore a lightweight eye-tracking device (PT mini) – this was for technical pilot reasons only and the eye-tracking data was not used in our study. Participants completed the tasks where they classified faces as either human or android, and

finished with a post-test questionnaire. The entire study took roughly 30 minutes.

Tasks

Our experiment consisted of two tasks: 1) classification accuracy priority, and 2) classification speed priority. In task 1, participants were shown a face for three full seconds, after which they were asked to classify it and were verbally asked post-stimulus questions before being shown the next face. The order of faces was counterbalanced using an incomplete Latin square. This design enaled the person to concentrate on the face and not to feel rushed in their decision making. In task 2, participants were shown the same faces (with a different counterbalanced order), but were asked to classify them as quickly as possible while the face was shown, with the response time being digitally recording. The rest of the presentation style was the same except there were no post-stimulus questions here.

In both cases, faces were shown at random locations on the screen, and a blank screen with a fixation cross in the center was placed between faces (during questions) to minimize cross-over effect. Further, in both cases the participant held a mouse in their hand with thumbs on the two buttons, and use these to classify the faces by pressing one of them (left for human, right for android).

Instruments

We compiled a database of faces, consisting of ten android and ten human faces, with half of each category being female. Faces were selected as much as possible to have a neutral expression and to be fully front-facing. Figure 1 shows faces used and provides source attributions. All images were scaled to 324x386 for consistency across faces, which was 3.4" x 4.5" on our screen. For the post stimulus verbal questions, for task 1, we asked the participant to rate how "eerie" the image was on a scale of 1-7.

The post-test questionnaire collect basic demographics, and included the Individual Difference in Anthropomorphism Questionnaire [24] to measure the participant's disposition toward anthropomorphism.

Results

Based on task 1 results, overall androids were more likely to pass for human (57%) than android (43%), and humans were uncommonly mistaken for androids (10%), $\chi^2(1)=38.87$, $p<.001$, and $\chi^2(1)=2.86$, $p<.1$ for android faces only (Table 1). Despite this result, however, there were marked differences in how the faces were prosessed.

We used one-taled, paired t-tests to further compare the results on android and human faces. On average, participants classified human faces faster (M=1.66s, SE=.14) than android faces (M=2.00s, SE=.14), (t_{13}=-4.16, $p<.01$, d=.7), reported that human faces were less eerie (M=2.21, SE=.33) than android faces (M=2.57, SE=.36, t_{13}=1.87, $p<0.05$, d=.28), and participants were found to have a lower error rate for classifying humans (M=.12, SE=.06) than androids (M=.51, SE=.07), (t_{13}=3.77, $p<.01$, d=1.65).

We performed correlation tests between the IDAQ (disposition toward anthropomorphism) questionnaire answers and other results, but all results were nonsignificant ($p>.50$). In addition, no effect of participant gender or face gender was found on any measure ($F<1$).

Table 1 – cross tabulation of how human and android faces were classified, $\chi^2(1)=38.87$, $p<.001$.

Count

		classified as		
		human	android	Total
face	human	126	14	140
	android	80	60	140
Total		206	74	280

Figure 1 – Face images used in our study, human faces on the left and android faces on the right. The human faces are extracted from the FEI face database (http://fei.edu.br/~cet/facedatabase.html). We compiled the android faces through sources available online. From left to right, top row first: Hanson Robotics' Philip K. Dick, Bina 48, Jules (http://www.hansonrobotics.com/), ATR Geminoid and Geminoid F. Second row, FaceTeam FACE robot [18], JST ERATO Asada and Kokoro CB2 , Neurobiotics Alissa, KITECH Ever-2, National Taiwan University Robot (unnamed).

DISCUSSION

Our results show that people classified android faces as human faces at the confidence level of 90% (or α=.1). At least for the simplified problem space of static images, we believe this demonstrates that android faces are doing quite well in terms of believability. Further, as expected, participants classified human faces faster than android faces, had a lower error rate, and found them less eerie. This shows that, even when android faces may pass for human, there are elements of visual processing and response that can highlight the differences between human and android faces. This initial pilot result lends support to our visual processing approach, and highlights how it can be applied to gain insight into believability of faces. For example, that through "expectancy violation" faces that have issues will take longer to process, and will be more ambiguous.

As the android faces were seen as being more eerie than the human faces, post-hoc we performed correlation tests between eeriness and the error rate and response time, to see if eeriness may predict the other factors. Unfortunately these tests were not significant. We believe that it will be important to continue to investigate how eeriness relates to visual processing and believability of faces.

The lack of results relating to disposition to anthropomorphize was surprising. Given the very poor results (t<1 in most cases, illustrated in Figure 2), we do not feel that this would become significant with more participants with our current setup. However, it is difficult to determine if the affect did not exist, if our sample size was too small, if other factors were larger than tendency to anthropomorphize, or if our IDAQ questionnaire did not measure it well, and so we encourage further inquiry in this area. Similarly, we suggest further inquiry into the effect of gender (both the person's [4] and the android's) on face believability.

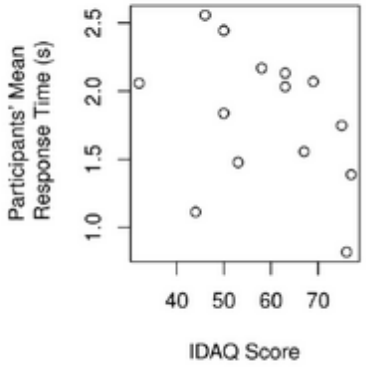


Figure 2 – Participant mean response time (s) against tendency to anthropomorphize (IDAQ score). As shown, there was little relationship found.

LIMITATIONS

A key limitation of this study was the small sample size. Further work in the area must address this to find results that are more generalizable.

The primary purpose of this paper is to explore the visual processing approach to considering the believability of android faces, and so our primary limitation is the small scope of the work, we hope that this direction continues to be developed, and stronger android-centric visual processing theories can be developed to better inform design. From this, we hope that further studies will continue to be conducted to unpack the complexity of android face believability. For example, one element of our theory which we did not address yet is the gaze element of facial processing - how people fixate and process a face.

One facet which must be addressed for continuing work in this area is the development of a standardized face database. Our results are deducted from only ten faces of android faces. While we attempted to maintain uniform lighting, angle, and size across them, taking more care to develop such a database with more faces would provide an excellent benchmark for researchers to compare against. Such a database could include meta-data such as the race, supposed age, and gender of the face.

CONCLUSION

Understanding why a particular android's face is believable, and what can be done about it, is an important challenge for android science. While this is a large challenge, in this work we provided a new angel on the problem, explicitly looking to visual processing knowledge to understand how people are viewing faces. Using this, we have discovered how recognition time and error rate, as well as perhaps perception of eeriness, can all be indicators of believability of an android face as human. In addition, through our exploration we have highlighted various other future directions for explorations in this area.

REFERENCES

1. Bartneck, C. et al. 2009. Does the Design of a Robot Influence Its Animacy and Perceived Intelligence? International Journal of Social Robotics. 1, 2 (Feb. 2009), 195–204.
2. Bartneck, C. et al. 2007. Is the uncanny valley an uncanny cliff? Proceedings - IEEE International Workshop on Robot and Human Interactive Communication (2007), 368–373.
3. Barton, J.J.S. et al. 2006. Information processing during face recognition: the effects of familiarity, inversion, and morphing on scanning fixations. Perception. 35, (2006), 1089–1105.

4. Bate, S. et al. 2010. Socio-emotional functioning and face recognition ability in the normal population. Personality and Individual Differences. 48, 2 (2010), 239–242.
5. Blow, M., Dautenhahn, K., Appleby, A., Nehaniv, C. L., & Lee, D. (2006, March). The art of designing robot faces: Dimensions for human-robot interaction. In Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction (pp. 331-332). ACM.
6. Blow, M., Dautenhahn, K., Appleby, A., Nehaniv, C. L., & Lee, D. C. (2006, September). Perception of robot smiles and dimensions for human-robot interaction design. In Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on (pp. 469-474). IEEE.
7. Bombari, D. et al. 2009. Featural, configural, and holistic face-processing strategies evoke different scan patterns. Perception. 38, (2009), 1508–1521.
8. Cheetham, M. et al. 2013. Category Processing and the human likeness dimension of the Uncanny Valley Hypothesis: Eye-Tracking Data. Frontiers in psychology. 4, March (2013), 108.
9. Chen, H. et al. 2010. Crossing the "uncanny valley": adaptation to cartoon faces can influence perception of human faces. Perception. 39, 3 (2010), 378–386.
10. Epley, N. et al. 2007. On seeing human: a three-factor theory of anthropomorphism. Psychological review. 114, (2007), 864–886.
11. Epley, N. et al. 2007. On seeing human: a three-factor theory of anthropomorphism. Psychological review. 114, (2007), 864–886.
12. Eyssel, F. and Hegel, F. 2012. (S)he's Got the Look: Gender Stereotyping of Robots1. Journal of Applied Social Psychology. 42, 9 (Sep. 2012), 2213–2230.
13. Hanson, D. et al. 2005. Upending the Uncanny Valley. Proceedings of the national conference on artificial intelligence. 20, 4 (2005), 24–31.
14. Hills, P.J. et al. 2013. First fixations in face processing: The more diagnostic they are the smaller the face-inversion effect. Acta Psychologica. 142, (2013), 211–219.
15. Ho, C. et al. 2008. Human emotion and the uncanny valley: a GLM, MDS, and Isomap analysis of robot video ratings. … conference on Human robot …. (2008), 169–176.
16. Ishiguro, H. 2007. Android Science - Toward a new cross-interdisciplinary framework -. Robotics Research: Results of the 12th International Symposium ISRR. Springer, 2007 (2007), 118–127.
17. Kitade, T. et al. 2013. Understanding suitable locations for waiting. ACM/IEEE International Conference on Human-Robot Interaction (2013), 57– 64.
18. Mazzei, D. et al. 2012. HEFES: An Hybrid Engine for Facial Expressions Synthesis to control human-like androids and avatars. Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics (2012), 195–200.
19. McCloud, S. 1994. Understanding Comics.
20. Miellet, S. et al. 2010. Investigating cultural diversity for extrafoveal information use in visual scenes. Journal of vision. 10, (2010), 21.
21. Morewedge, C.K. 2009. Negativity bias in attribution of external agency. Journal of experimental psychology. General. 138, (2009), 535–545.
22. Mori, M. et al. 2012. The uncanny valley. IEEE Robotics and Automation Magazine. 19, 2 (2012), 98–100.
23. Peterson, M.F. and Eckstein, M.P. 2013. Individual differences in eye movements during face identification reflect observer-specific optimal points of fixation. Psychological science. 24, (2013), 1216– 25.
24. Peterson, M.F. and Eckstein, M.P. 2012. Looking just below the eyes is optimal across face recognition tasks. Proceedings of the National Academy of Sciences of the United States of America. 109, (2012), E3314–23.
25. Schwaninger, A. et al. 2006. Chapter 18 Processing of facial identity and expression: a psychophysical, physiological, and computational perspective. Progress in Brain Research. [26] Waytz, A. et al. 2010. Who Sees Human? Perspectives on Psychological Science. 5, (2010), 219–232.

C.3 Sample Paper with High Error Frequency

Using Fake Cursors to Secure On-Screen Password Entry

Anonymous Author 1
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 2
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 3
Institution of Anon.
Anon. Address
email@anon.ano

ABSTRACT

In this paper, we present a concept using fake cursors to disguise on-screen password entry. We performed two user studies with different amounts of dummy cursors and differently colored cursors. The results show that dummy cursors significantly improve security. At the same time, decrease in performance is kept within an acceptable range. Depending on the required degree of security, the studies favor 8 or 16 differently colored cursors as the best trade-off between security and usability.

INTRODUCTION

Password entry is a ubiqitous task. In many instances, the user has to authenticate in a public or semi-public setting like internet cafés or office environments, exposing the password to onlookers. On-screen keyboards are often used to minimize the possibility of losing the password due to keyloggers and other malicious software. For instance, this is commonly used by online banking websites. They enforce the use of virtual keyboards or keypads to input the secret credentials. While being more secure against keyloggers and the like, this approach is highly vulnerable to shoulder surfing attacks, that is, an attacker observing the input from a nearby position. It's almost impossible to hide the input as this would mean covering a large portion of the screen space.

One of most common solutions to this problem is adding overhead to the input to make it hard to follow. A famous example is the spy-resistant keyboard by Tan et al. [5] which uses an indirect input method to make on-screen keyboard use more secure. Unfortunately, indirect input makes the interaction with such a system quite slow. Other researchers add overhead in the form of fake input (e.g. adding additional digits to a password) like Vibrapass by De Luca et al. [1]. Graphical authentication systems like the one presented by Wiedenbeck et al. [7] use indirect input in the form of distracting icons or images to hide the input. The randomness introduced with these systems makes them slower and significantly reduces memorability. Finally, some research focusing on using additional hardware to make the input invisible to an attacker [3] or dislocate the input from the terminal [4].

We propose a shoulder surfing resistent input method using multiple fake cursors. The idea is inspired by Ninja cursors [2]. In their work, the authors propose using several

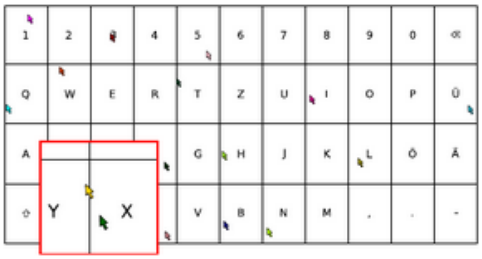


Figure 1. 16 colored mouse cursors in the user study. The red outlined area was created for the screenshot for better visibility.

concurrent cursors that move in the exact same way to quickly reach objects on big screen spaces. As opposed to Ninja cursors, in our system, only one cursor performs the actual input while the other cursors act as distraction for an attacker. That is, they do not move in line with the genuine cursor.

Quiet recently, Watanabe et al. [6] presented a demo of a similar approach to secure PIN-entry. The main differences to their work are that we use an intelligent cursor algorithm instead of pre-recorded cursor movements; that we introduced coloring to improve the usability of the system; and that we performed extensive evaluations of the approach to prove its appropriateness to secure on-screen password entry.

The results of the studies show that, depending on the required level of security, using 8 or 16 differently colored cursors provides a good trade-off between security and usability. In either case, he approach significantly improves shoulder surfing resistance of on-screen password entry.

CONCEPT

The main idea is using a specific number of fake cursors that hide the input on the on-screen keyboard from onlookers. Since the fake cursors move differently from the active cursor, users can identify it while attackers have difficulties to do so. Figure 1 shows an on-screen keyboard with 16 colored mouse cursors, 15 of them fake cursors. The active cursor has a random color.

The fake cursors move with respect to several rules: a) They use bezier-like paths to a randomly selected target on

1

the keyboard as strait movements would give them away. b) When the active cursor changes it's direction by more than 90◦, the distraction cursors pick new targets using similar angles. c) They never leave the keyboard and are always located at a target when the active cursor is. Off-keyboard cursors would easily be identified as fake.

By itself, on-screen keyboard-entry is already slow. Adding fake cursors does further reduce the input speed but at the same time significantly improves its resistance to shoulder surfing attacks. Such an approach is not meant for contexts with several authentication sessions per day but for systems with high security demands like online banking.

Threat Model and Theoretical Security

The approach was designed to be resistant against shoulder surfing attacks. For a successful attack, the attacker needs to observe all cursors or immediately identify the active cursor, no visual feedback of clicked buttons is given. The most promising attack on the system is video-based (combination of screen and mouse recording). This would enable comparing the mouse movement to the movement of the cursors on screen and therewith identify the active cursor.

Attack-resistance is highly depending on the strategy that the user employs to identify the active cursor as described later in this paper. If fast unnatural movements are used, the current algorithm of the fake cursors does not keep pace and the active cursor becomes obvious.

PRE-STUDY:

We conducted a pre-study with a simple pointing task using a repeated measures factorial design with two independent variables: *Cursors* (4, 8, 16, 24) and *Color* (differently colored cursors or all white). We did this to test whether color makes it easier to keep track of the active cursor. To minimize learning effects, we used a 2x4 Latin square design resulting in eight cases. The main goals of the pre-study were: 1) First insights on the usability using different numbers of cursors. 2) Identify search strategies for the active cursor.

PROCEDURE AND PARTICIPANTS

The study took place in an isolated room at our premises. For each combination of the independent variables, the participant had to find the active cursor and then use it to click two targets, located on the screen, in a predefined order. The location of the targets was not varied. Three trials were allowed to correctly perform each task. If successful or after failing three times, the next task started. After each trial, the cursors (including the active one) were randomly arranged on the screen. That is, active cursor had to be found again.

The task order was automatically assigned by the prototype based on the user ID. Each participant performed each possible subtask twice resulting in 16 tasks per user. At the end of the study, the participants were asked to fill out a questionnaire collecting demographics and qualitative data.

We recruited 16 participants for the study. That is, the Latin square design was applied two times. The average age was(range: 19-51). Five participants were female. Participants received a 5 Euro voucher for an online shop.

Results and Discussion

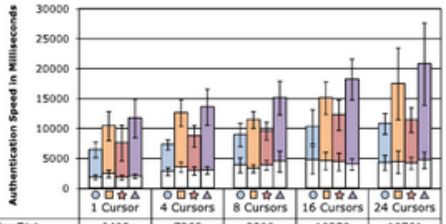
Search strategies are not only interesting from a point-of-view of usability but they are also related to security. A bad strategy can influence weather an attacker can identify the active cursor or not. The analysis of the questionnaire and the video material revealed two main strategies: 1) Eight participants moved the mouse cursor to the border of the interaction area to identify it. This strategy can influence the security since the fake cursors do not behave this way. 2) Five participants moved the mouse in small shapes (e.g. waves). This is notplicated by the algorithm and thus, based on good hand-eye coordination, the active cursor can be identified.

Performance-wise, the most interesting results is on the time required to finish the task, to identify the active cursor and to keep track of it. Keeping track refers to the fact that the user has to be able to follow the cursor and not loose track of it. The fastest task completion time was achieved with four colored cursors (M=4.9s), the slowest one with 24 white cursors (M=10.7s). A 4 x 2 (*Cursors* x *Color*) within participants analysis of variance of task completion time revealed a highly significant main effect for *Cursors* ($F_{3,45} = 7.739$, $p < .001$). Post-hock tests revealed a significant difference between Cursors levels 4 (M=5.1s) and (M=8.9s) as well as 4 and 24 (M=10.7s; both $p < .05$).

We considered the time to hit the first target as the timequired to find the active cursor and the time from the first to the second (last) target as the performance of keeping track of the active cursor. Analyzing the data based these assumptions revealed an interesting finding. While the color did neither significantly influence the overall performance nor the time to find the active cursor, it significantly influenced the tracking performance. Using the same ANOVA, we could identify a significant main effect of *Color* ($F_{1.0,15.0} = 6.608$, $p < .05$). Keeping track of a coloredtive cursor (M=1.9s) was significantly easier than of a white cursor (M=2.5s). In addition, the positive influence of *Color* increases with the amount of cursors.

MAIN STUDY

The main study was firstly conducted with 39 people and then repeated with 20 of those participant's. The first part will not be presented here and was only done to have access to trained users for the second iteration.



	1 Cursor	4 Cursors	8 Cursors	16 Cursors	24 Cursors
Color Dict (search)	6408 (1830)	7268 (2791)	8900 (3858)	10250 (4722)	10761 (4262)
Color No Dict (search)	10392 (2421)	12592 (3522)	11408 (3308)	15071 (4566)	17435 (4371)
No Color Dict (search)	7568 (1795)	8685 (2905)	9540 (3874)	12237 (4414)	11407 (4182)
No Color No Dict (search)	11666 (2000)	13545 (2979)	15070 (4480)	18184 (3985)	20728 (4693)

Figure 2. Average overall authentication speed. White bars and brackets indicate the time required to find the active cursor.

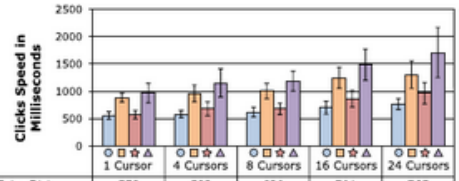
We used a repeated measures factorial design with three independent variables: *Cursors* (1, 4, 8, 16, 24), *Color* (yes, no) and *Password* (dictionary, non-dictionary). Passwords consisted of six characters. Non-dictionary passwords were randomly created containing upper case and lower case letters as well as digits and special characters. Dictionary passwords were selected from a list of most commonly used words in German, the mother language of all participants. Please note that a new level was introduced to *Cursors*. Level 1 was required to have a baseline to compare the performance of the system. To minimize learning affects, we used a 2x5 Latin square design resulting in ten cases. *Password* was randomized. An example with 16 colored cursors is shown in figure 1. Having 20 participants allowed for two repetitions of the Latin square design. All participants were familiar with the system from the first study iteration.

Procedure and Participants

The study took place in an isolated room at our premises. It was filmed with a hi definition camera from the right side of the user, recording both the mouse and the screen. The purpose of the camera was: 1) To identify strategies and usability issues like in the pre-study. 2) To use the video material for a security analysis after the study.

At the beginning, the password task was explained to the participants. For each combination of the independent variables, they had to input two passwords (dictionary and random). When it was not correct, the next task didn't start and the user had to fix it by deleting the input pressing "backspace" on the virtual keyboard. The start position of the cursors was randomized after each task.

The participants received a unique list from the experimenter containing their passwords in the order of the experiment. It should be noted here that the users did not get the same passwords that they used in the first iteration



	1 Cursor	4 Cursors	8 Cursors	16 Cursors	24 Cursors
Color Dict	558	582	620	701	765
Color No Dict	883	962	1005	1248	1305
No Color Dict	573	681	680	867	964
No Color No Dict	968	1153	1186	1484	1709

Figure 3. Average time needed between two button presses.

of the study. At the end of the study, the participants were asked to fill out a questionnaire collecting demographics and qualitative data.

The 20 particicpants had an average age of 25 (range: 16 - 32). Nine of them were female. Again, 5 Euro vouchers for an online shop were given out as incentives.

Results and Discussion

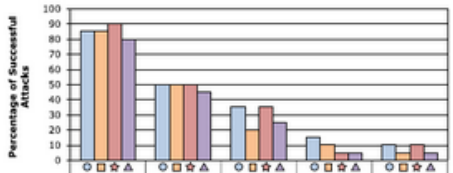
All statistical results are based on a 5 x 2 x 2 (*Cursors* x *Color* x *Password*) within participants ANOVA. Errors were very low across all conditions and thus error rates will not be discussed in this paper.

Authentication Speed

We diffferentiated three interaction times: overall (timequired for the whole authentication), search (time to findactive cursor, i.e. before the first click) and click (average time between all button clicks). "Overall time" presents realistic measure of how long authentication takes. "Search time" provides an approximate value of how long it takesfind the active cursor. Finally, "click time" indicates how easy it is to keep track of the active cursor.

The results of overall and search time are depicted in figure 2. It shows that while it takes longer to find the active cursor with more cursors, *Color* and *Password* did not influence that time but it is much lower for one cursor. It is interested to note that search time only slightly rises with the amount of cursors. There was a highly significant main effect for *Cursors* ($F_{2.017,38.325} = 16.858$, $p < .001$). Post-hoc tests showed significant differences between 16/24 cursors and all other levels of *Cursors* (all $p < .05$) and no significant difference between 16 and 24 cursors.

For overall time, effects of all three independent variables can be observed. One cursor using dictionary passwords was the fastest input method (M=6.4s). 24 cursors without color and random passwords was the slowest (M=20.7s). We found highly significant mane effects for *Cursors* ($F_{2.557,48.589} = 15.794$, $p < .001$) and *Password* ($F_{1.0,19.0} = 80.05$, $p < .001$) and a significant main effect for *Color* ($F_{1.0,19.0} = 9.093$, $p < .05$). Post-hoc tests showed that colored cursors were faster ($p < .05$) and dictionary passwords were faster than random passwords ($p < .001$).



	1 Cursor	4 Cursors	8 Cursors	16 Cursors	24 Cursors
Color Dict	85	50	35	15	10
Color No Dict	85	50	20	10	5
No Color Dict	90	50	35	5	10
No Color No Dict	80	45	25	5	5

Figure 4. Percentage of successful sholder surfing attacks on the system. While 16 and 24 cursors are significantly more secure than 1, 4 and 8, there is no apparent difference between them.

Additionally, there were significant differences between 16/24 cursors and all other numbers and no significant differences between 16 and 24 cursors.

The results for click time are shown in figure 3. We found highly significant main effects for *Cursors* ($F_{2.502,47.546}$ = 26.183, $p < .001$), *Password* ($F_{1.0,19.0}$ = 15.946, $p < .001$) and *Color* ($F_{1.0,19.0}$ = 91.231, $p < .001$). Post-hoc analysis results are similar to overall time.

We again asked the participants which strategies they used to find the active cursor. 13 of them used the strategy of moving the cursor to one of the borders of the interaction area. Two users based finding the cursors solely on performing specific shapes with the cursors. The final five used both strategies depending on the situation. No new strategies were found.

Security

The video material was used to perform shoulder surfing attacks. The attacker was highly familiar with the system. He was allowed to watch the input once to steal the password. The result was compared to the original password using the Levenshtein distance, a measure of similarity between two strings (operations necessary to get from one string to the other). Distance "0" indicates a correct guess. Overall, the analysis of all video material took the attacker one full day.

Figure 4 shows the results for the security analysis. In the worst case (one cursor, no color, dictionary), 90% of passwords were successfully shoulder-surfed. The most secure instances were 16 cursors (no color, dictionary and random) and 24 cursors (color, random and no color, dictionary) with 5% success rate. There is no obvious improvements using 24 cursors compared to 16 cursors. Additionally, color does not seem to influence the security of the system as well.

The Levenshtein distance produces parametric data (ratio) and thus allows for using parametric significance tests. We found a highly significant main effect for *Cursors* ($F_{4,76}$ = 72.863, $p < .001$). No other main effects and no interaction effects could be identified. Post-hoc tests revealed (highly) significant differences between all levels of *Cursors* with the exception of 16 and 24 cursors (p = 1.0).

There was no influence of strategies to find the active cursor on the amount of stolen passwords per user, this was most probably since the number of users that did not use the border strategy, at least to some extent, was quite low. However, we could observe awareness of the fact that the strategies might influence the security. For instance, one user stated that "*I avoided moving the cursor to the border as I think this would negatively influence the security of the system ...*".

Best Combination

The results shows that the advantage of colored cursors kicks in after the active cursor has been identified. That is, it is easier to keep track of the active cursor if all cursors are diferently colored. On the other hand, security was not influenced by colors. We argue that depending on the required degree of security, 8 or 16 colored cursors are the best trade-off between security and usability. For high security contexts like online banking, 16 cursors present the best solution. The very low error rates across all conditions support this claim.

This is supported by the results of the questionnaire, in which I asked the participants to rank the different combinations with respect to the best security-usability trade-off. Summed up, 13 participants ranked 8 and 16 colored cursors as the best trade-off (6*16, 7*8). One participant chose 24 colored cursors. In addition, on Likert scales from 1 (no agreement) to 5 (full agreement), 19 participants either fully (14) or partially agreed (5) that the colors improve ease-of-use. 13 participants either fully (6) or partially agreed (7) that the approach improves security (no one disagreed)

CONCLUSIONS AND FUTURE WORK

We presented a system using fake cursors to hide password entry on on-screen keyboards. Two user studies showed good usability properties and a significant increase in security. The best trade-off between usability and security was achieved with 8 and 16 differently colored cursors respectively.

Even though the participants were trained, they cannot considered experts. We could not test for learning effects but believe that there is room for improvement. For instance, one study participant mentioned that he "*did it several times now and it gets easier every time*". An open question is therefore how the system performs at long-term use. We plan to conduct a long-term web-based study to answer this question. For instance, it will be interesting to find out if users become significantly faster after long-term use and if they develop more advanced search strategies.

We have a few ideas on how to avoid the border strategy and improve authentication speed. The most promising one is to assign a fixed color or a fixed start location to the active cursor. This way, the cursor can be identified by simply looking for the cursor with that color or at the specific key. We will conduct further studies to find out a) if this approach has the potential to improve search time and b) if this way, other strategies for finding the cursor become obsolete. We argue that this will at the same time reduce authentication speed and make the current algorithm more efficient as unusual behavior is not anymore necessary to identify the active cursor

REFERENCES

1. De Luca, A., von Zezschwitz, E., and Hussmann, H. Vibrapass: secure authentication based on shared lies. In Proc CHI '09 (2009), 913–916.
2. Kobayashi, M., and Igarashi, T. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In Proc. CHI '08 (2008), 949–958.
3. Sasamoto, H., Christin, N., and Hayashi, E. Undercover: authentication usable in front of prying eyes. In Proc. CHI '08 (2008), 183–192.
4. Sharp, R., Scott, J., and Beresford, A. Secure mobile computing via public terminals. In Proc. Pervasive '06 (2006), 238–253.
5. Tan, D. S., Keyani, P., and Czerwinski, M. Spy-resistant keyboard: more secure password entry on public touch screen displays. In Proc. OzCHI '05 (2005), 1–10.
6. Watanabe, K., Higuchi, F., Inami, M., and Igarashi, T. CursorCamouflage: multiple dummy cursors as a defense against shoulder surfing. In SIGGRAPH Asia '12 Emerging Technologies (2012), 6.
7. Wiedenbeck, S., Waters, J., Sobrado, L., and Birget, J.-C. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In Proc. AVI '06 (2006), 177–184.

C.4 Error Classification Paper Post-categorization

Twist and Pulse: Ephemeral Adaptation to Improve Icon Selection on Smartphones

Anonymous Author 1
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 2
Institution of Anon.
Anon. Address
email@anon.ano

Anonymous Author 3
Institution of Anon.
Anon. Address
email@anon.ano

ABSTRACT

The concept of ephemeral adaptation was introduced to reduce visual search time in GUI menus, while preserving spatial consistency and minimizing distraction. We extend this concept to the visual search of app icons on smartphones, to speed up launching apps from a homescreen. We created ephemeral highlighting effects based on preattentive visual properties including size, orientation, color, opacity and blur. We then conducted multiple cycles of rapid design and informal evaluation from which Twist (icon rotates back and forth) and Pulse (icon grows and shrinks) were the most promising effects. An experiment comparing these two effects to a control condition showed that these effects improve search time performance by 8-10% and Pulse is subjectively preferred to the control condition.

INTRODUCTION

The growing varity and number of mobile applications, or *apps*, have dramatically expanded the functionality of smartphones. Not surprisingly, the number of apps installed by users has rapidly increased in recent years [2]. This growth has resulted in several usability problems, one of which is visually searching for the desired icon in order to launch an app [2]. This problem is exacerbated because the icons that are commonly used for these apps are often complex and lack adequate discriminatory features [3].

There is, however, someregularity to the patterns of app usage that can be leveraged in order to speed up access: only a small proportion of apps are used frequently, and different apps are used in different contexts [2,6,8]. These patterns have enabled successful prediction of the most probable app selections [6,8].

To address the problem of accessing app icons, researchers have tried to beverage these predictive models by designing adaptive interfaces that speed up access of the predicted icons. Two common techniques for visually distinguishing the predicted icons from other distractor icons have been studied: static highlighting using a bright surrounding halo [8] and copying the likely-to-be-used apps to a separate part of the interface [8]. None of these techniques have been shown to significantly improve icon selection performance. The main problem with static highlighting is that its permanence can negatively impact performance if the target icon is not predicted by the algorithm. In contrast, copying icons results in spatial inconsistency, which has been shown to confuse users [5,8].

In the context of menu selection, Findlater et al. addressed the problem of visual search by introduces a technique called *ephemeral adaptation* [4] , which involves adapting the interface and reverting it back to its normal state after a short period of time. Findlater et al. implemented this idea by first displaying the predicted items then gradually fading in the non-predicted items, maintaining spatial consistency throughout. They used the temporal dimension to atract attention to predicted items and showed that this technique results in faster selection times than static highlighting.

Inspired by this idea, we extended and evaluated the concept of ephemeral adaptation in the context of icon visual search on smartphones, which is a more complicated visual task thn searching items from a menu. Menu items are dominated by their text labels, whereas apps are dominated by their large icons which vary considerably in complexity. We leverage the human visual system by using preattentive roperties including size, orientation, color, opacity and blur to make predicted icons more salient.

The contribution of this paper is as follows. To the best of our knowledge, we are the first to extend the ephemeral technique to visual icons. We explore the design space of ephemeral highlighting for visual icons through rapid iterative design and evaluation cycles. In doing so, we document which preattentive visual properties work well; and which do not. The two most promising ephemeral visual effects that emerged are Twist (icon rotates back and forth) and Pulse (icon grows and shrinks), as shown in the Figure 1. Our experiment shows that these two effects significantly improve search performance over a control condition, independent of the number of pages of icons and the accuracy of the prediction algorithm used. Pulse is also preferred subjectively.

EPHEMERAL HIGHLIGHTING EFFECTS

Design Considerations

The design goals for our ephemeral highlighting effects

1

Figure 1. Direct and indirect ephemeral highlighting effects applied to the predicted weather icon (top right icon). The other three icons are non-predicted. Each effect is labelled with its corresponding preattentive visual property (top), and the name we used to describe it to participants (bottom, in quotes). The arrows indicate how the direct highlighted icons change over time.

were to attract attention to predicted icons, to maintain spatial consistency, and to minimize distraction so as to be suitable for everyday use. Therefore, our designs make predicted icons stand out on preattentive visual properties such as size, orientation, color, opacity and blur [7]. Our exploration of the design space was intended to be thoughtful but also opportunistic, starting with a reasonable set of preattentive properties and varying their parameters, rather than exhaustive of all possibilities. Figure 1 shows a sample set of designs. The nature of the individual properties means that we either *directly* transform predicted icons to make them more salient, or we "dim" all the non-predicted icons to make the predicted ones stand out—an approach we refer to as *indirect* highlighting.

Each preattentive property includes a number of parameters that further expand the design space. For instance, we considered a range of angles for Rotate, from 10° to 180°, and explored different sizes for Shrink, from 10% larger to 50% larger. Additionally, we experimented with the opposite effect (Grow, not shown in Figure 1) with sizes ranging from 10% to 50% smaller. For indirect highlighting effects, we varied the alpha value (Transparency) of the non-predicted icons from .1 to .5, and the Gaussian blur radius from 2 to 10.

The ephemeral nature of these highlighting effects requires the interface to return to it's normal state after a certain period of time. The total duration of the effects is therefore a critical parameter. Findlater et al. explicitly compared 250ms and 500ms durations, and found the latter to be th most effective. However, 500ms was often not enough for indirect highlighting effects to be perceived; for these effects, we increased the total duration to 1000ms.

To make the effects more visually apealling we added smooth transitions between the highlighted state and the normal state. In the case of indirect highlighting like Greyscale, the colors of the non-predicted icons simply fade in slowly. For Size and Orientation, the predicted icons are animated back to their normal appearance (hence the names Shrink and Rotate). We also tried adding smooth transitions at the beginning of the effects to reach the highlighted state. For example, Twist and Pulse are variants of Size and Orientation with a continuous animation from normal to highlighted then back to normal. There is therefore no abrupt visual change when a new page of icons is loaded. We explored different durations for these transitions between states, from 100 to 1000ms. We found that a delay before transitioning was useful for the indirect highlighting effects as it gave more time to find the highlighted icons; however, this pause was found superfluous and even annoying for the direct effects.

We note that the use of transitions with the direct effects meant that highlighted icons were perceived as being in motion. While motion is one of the best ways to attract attention [1], we were interested to see if it would be too annoying for this task, and whether indirect techniques provide a better balance between.

Preliminary Feedback

We iteratively refined our designs via informal evaluation and pilot testing with 16 smartphone users. Participants were asked to select a target icon among three pages of apps on a smartphone, 20 icons per page; in total, nine icons were highlighted, only one highlighting effect was used at a time. The goal was to identify the most promising effects, and to adjust their parameters (e.g., total duration, transition duration, size, angle). The key findings are summarized below.

Blur was discarded early: the in-focus predicted icons could hardly be told apart from non-predicted blurred ones, unless the latter were degraded to unrecognizable color blobs.

Color and Opacity were more effective than Blur, but participants sometimes had trouble identifying which icons were highlighted, especially when several were highlighted on the same page at once. Color was particularly problematic: as one of our participants pointed out, *"some of the icons don't have a color to begin with!"* Additionally, all indirect highlighting effects share the same weakness: they degrade the appearance of non-predicted icons, making them less recognizable. If the algorithm's prediction is wrong, it is harder for users to finding the icon they are looking for: the user must wait for the ephemeral effect to end and the non-predicted icons to avert to their normal appearance, diminishing the benefits of adaptive highlighting.

The effects involving some form of motion were found to be the most effective: Shrink, Rotate, Twist and Pulse. We learned that increasing the size of an icon is more likely to attract attention than decreasing it because it visually corresponds to an object coming towards the observer. Most participants expressed a preference for Pulse over Shrink, finding it *"more aesthetic"* and having *"an organic feel"*. Twist was unanimously well received, and described as *"cute"*, *"playful"* and even *"chic"*. We therefore selected Twist and Pulse for further study. The key final parameters were as follows: a max size of +25% for Pulse, a max angle of ±15° for Twist, and a transition duration of 450ms plus a short 100ms delay at the outset for both (Todo: talk more about the delays here!)

EXPERIMENT

The main goal of this experiment was to determine if Twist and Pulse could improve performance for selecting icons, by reducing the selection time and/or the number of errors. We also investigated two factors: (1) the number of pages of apps users have to search through, in order to assess the impact of app volume; and (2) the accuracy of the underlying prediction algorithm, as adaptive interfaces are known to cause frustration when the accuracy of the prediction is inadequate [5].

methodology

Participants & Apparatus: The study included 12 participants (mean age 25, 4 females), all frequent smartphone users. On average, participants had 50 apps across 3.8 pages on there own phones (1 min, 8 max pages). The custom experiment application ran on two Google Nexus 5 devices. App icons were retrieved from Icon100, an online database (www.icon100.com), while labels for the icons were sampled from a corpus of neutral nouns.

Task: The expearmint consisted of a sequence of icon selection trials. In each trial, participants were shown a target icon consisting of an image and a label. Tapping once on the screen brought users to the first page of 20 icons, arranged in a 5x4 grid. Participants could navigate through the pages by swiping. Once they clicked on an icon, the target for the next trial was displayed. If an icon other than the target was selected, the application provided feedback and proceeded to the next trial. The system reported a timeout when a trial hit 20 seconds, and moved on to the next trial. To generated a sequence of icon selections, target icons were randomly sampled from a Zipfian distribution, as in Findlater et al. [4]. For 40 trials, we sampled 20 unique target icons with relative frequencies 8, 5, 3, 3, 2, 2, 2, 2, 2, 1 ... 1 (Zipfian $R^2 = .93$).

Pages, Accuracy & Prediction Algorithm: To represent a range of possible configurations, we chose to try both 3 and 6 pages of apps, roughly equivalent to a average and a large number of apps on a smartphone. To determine realistic accuracy levels, we used Shin et al.'s comparison of various algorithms for predicting app usage on smartphones [8], which showed that the accuracy of prediction algorithms depends on the number of apps predicted. Based on our pilot study, we decided to highlight on average 3 icons per page, to insure that participants could perceive all of them. Therefore, 9 and 18 icons were highlighted for 3 and 6 pages, respectively. According to Shin et al., this corresponds to accuracy levels of 80% to 95%. We used these too values in our experiment.

Predicted icons were highlighted by a custom prediction algorithm, similar to the one used by Findlater et al. [4]. It combines three prediction strategies: most frequently used (MFU), most recently used (MRU), random selection. The predictions of the algorithm were randomly adjusted before the experiment to reach the desired accuracy levels*and random selection. The predictions of the algorithm were randomly adjusted before the experiment to reach the desired accuracy levels*, defined as the percentage of trials in which the target icon is among the ones predicted by the algorithm (and therefore highlighted with one of our effects).

Design: A 3-factor within-subject design was used: 3 highlighting effects (Twist, Pulse, Control—no effect) x 2 prediction accuracies (80%, 95%) x 2 number of pages (3, 6). Prediction accuracy and number of pages were fully counterbalanced across participants, while order of highlighting effect was determined by a Latin Square. Both icons and labels were randomly sampled for each combination for each participant. Each participant completed 480 trials (12 combinations x 40 trials) for a total of 5760 trials across all 12 participants.

Procedure: The experiment fit into a single 90-minute session. Participants first completed a demographic questionnaire before doing all 12 combinations of the three factors. In each combination, participants had a chance to practice, followed by 40 timed trials. Both completion time and error rate were recorded. Participants first seen the 6 combinations at one accuracy level before moving onto the combinations with the other accuracy level. Participants were informed that the prediction may occasionally be wrong, and they were told at the beginning of their second assigned accuracy level that the algorithm had changed and it could be better or worse than before. After each accuracy level, participants were given a questionnaire with subjective Likert scale questions. At the end of the study, compare and rank their overall preference and perceived performance for the three effects.

Results

Prior to the analysis, we removed all the trials that timed out. Out of 1920 total trials per highlighting effect, we removed 32, 30 and 27 timeouts corresponding to Control,

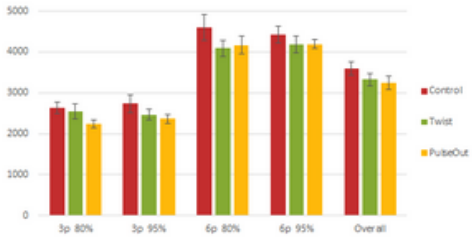


Figure 1. Selection times (ms) in each combination of accuracy and number of pages. Overall aggregates across all the data.

Twist and Pulse, respectively. Additionally, participants selected the wrong icon 39, 33 and 22 times for Control, Twist and Pulse, respectively. These trials were excluded as well, to avoid disadvantaging Control.

Speed: Selection time was measured as the time from tapping on the displayed target icon to selecting an icon. We ran a 3-way RM ANOVA on selection time. As Figure 2 shows, there was a significant mane effect of Highlighting Condition ($F_{2,22}$=10.60, p=.001, η^2=.491). Twist and Pulse were both significantly faster than Control (p<.01 and p<.05, Bonferroni corrected), but not significantly different from each other (p=.87). Twist was 8% faster than Control, while Pulse was 10% faster. As expected, there was a main effect of the number of pages ($F_{1,11}$=456.32, p<.001, η^2=.976), with trials for 6 pages being much longer than trials for 3 pages. However, there was no main affect of accuracy (p=.90), nor any interaction effects.

Subjective findings: We ran non-parametric Friedman tests on the Likert scale questions and Chi square on the comparative ranking questions. While we asked participants for their preference after each level of prediction accuracy, there was no significant difference based on accuracy. The comparative ranking at the end, however, did show a significant difference in users preference and their perceived performance of the highlighting effects ($\chi^2_{(2,\ 12)}$=18.19, p=.001, and $\chi^2_{(2,\ 12)}$=22.82, p<.001, respectively). 11/12 participants' preferred Pulse to Control, perceiving it as being faster; only 6/12 participants preferred Twist to Control despite 9/12 perceiving it as being faster.

DISCUSSION

The results of our proof-of-concept experiment demonstrate the potential of extending ephemeral adaptation for selecting app icons. Both Pulse and Twist were faster than Control by 8 to 10%—a speedup comparable to previous work in adaptive interfaces [4]. Interestingly, the two levels of accuracy did not impact the performance of the participants, which means that performance is not sensitive to having an imperfect prediction algorithm—users benefit from this technique even at 80% accuracy, which is achievable for 9 highlighted icons or more by using MRU or MFU icons [8]. In addition, there was no interaction between highlighting effect and the number of pages, which suggests likewise that our ephemeral techniques are not particularly sensitive to the number of pages of apps.

Although Twist and Pulse fared similarly in terms of selection time, participants expressed a preference for Pulse, judging it more effective for finding icons and less distracting than Twist, one possible explanation is that rotating the icon creates a motion blur stronger than scaling it up and down. Thus, even if both effects are equally effective at signaling which icons are highlighted, it may be harder to recognize an icon that twists than one that pulses.

While our results are promising, our experiment has some limitations, which come mostly from the simplified homescreen that we used. Recent mobile operating systems offer several mechanisms to help users launch apps, such as sorting them alphabetically, or allowing users to customize their homescreen by repositioning icons or grouping them into folders. Our technique should still provide benefits if apps are sorted alphabetically, or grouped into folders (the highlighting would follow the folder hierarchy). Howevr, if users move all their frequently used apps to the first page of their homescreen, the prediction algorithm should be modified to avoid highlighting too many icons (perhaps at most five icons per page). A cool way to reduce the number of highlighted icons per page is not to highlight the topmost frequently used apps, as users know their positions well and rely on muscle memory to launch them. Regardless, past work has found that a large proportion of users do not customize their homescreens [2], and could therefore benefit directly from our highlighting effects.

To conclude, I think that ephemeral adaptation is a promising technique to improve selection of app icons on smartphones. It is likely to provide benefits in other contexts that involve selecting an element among a large set of images such as photo galleries, album covers in music libraries, and the app panels featured in recent desktop operating systems.

REFRENCES

1. Bartram, L., Ware, C., and Calvert, T. (2003). Moticons: Detection, Distraction and Task. Int. J. Hum.-Comput. Stud. 58, 5, 515–545.

2. Böhmer, M. and Krüger, A. (2011). A Study on Icon Arrangement by Smartphone Users. Proc. CHI '11, 2137–2146.
3. Byrne, M.D. (1993). Using icons to find documents: simplicity is critical. Proc. CHI '93, 446–453.
4. Findlater, L., Moffatt, K., McGrenere, J., and Dawson, J. (2009). Ephemeral Adaptation: The Use of Gradual Onset to Improve Menu Selection Performance. Proc. CHI '09, 1655–1664.
5. Gajos, K.Z., Czerwinski, M., Tan, D.S., and Weld, D.S. (2006). Exploring the Design Space for Adaptive Graphical User Interfaces. Proc. AVI '06, 201–208.
6. Huang, K., Zhang, C., Ma, X., and Chen, G. (2012). Predicting mobile application usage using contextual information. Proc. UbiComp '12, 1059–1065.
7. Palmer, S.E. Vision science: Photons to phenomenology. MIT press Cambridge, MA, 1999.
8. Shin, C., Hong, J.-H., and Dey, A.K. (2012). Understanding and Prediction of Mobile Application Usage for Smart Phones. Proc. UbiComp '12, 173–182.

C.5 Post-test Questionnaire

Subject ID: _____

Date: _____

Time: _____

Post-test Questionnaire

1. Are you (check one):

☐ undergraduate ☐ graduate ☐ faculty ☐ other: _____

2. With which gender do you identify?:

☐ female ☐ male ☐ other ☐ prefer not to specify

3. Approximately how many conference or journal papers have you provided reviews for (check one)?

☐ 0 ☐ 1-5 ☐ 6-10 ☐ 11-15 ☐ 16+

4. Please circle the number that indicates how much you agree with the following statements (1 being "strongly disagree" and 5 being "strongly agree").

Statement	strongly disagree		neutral		strongly agree
a. Pointing out writing errors using the highlighting tool is easy.	1	2	3	4	5
b. Highlighting writing errors is time consuming compared to just reading and not highlighting.	1	2	3	4	5
c. Highlighting was sufficient for allowing me to point out the writing errors I found.	1	2	3	4	5
d. As an author, I would find it useful to have my writing errors highlighted by reviewers.	1	2	3	4	5
e. I would be willing to catch writing errors for authors errors using a tool like this in peer review sessions.	1	2	3	4	5
f. As an author, I would find it useful if my writing errors were highlighted AND classified.	1	2	3	4	5
g. As a reviewer, I would be willing to highlight AND classify errors for authors.	1	2	3	4	5
h. During the experiment, I highlighted every writing error I found.	1	2	3	4	5

Version 1.0 2015-Mar 03 Post-test Questionnaire page 1/2

5. Do you have any general comments or wish to elaborate on any of the above? If so, please write below.

_____

_____

_____

_____

_____

_____

C.6 Supplementary Data

C.6.1 Mixed-design ANOVA

This section shows a walkthrough of the analysis of variance for the experiment described in Chapter 4. Highlighted table cells indicate noteworthy results.

Mauchly's test of sphericity

Measure: ProportionFound

Within Subjects Effect	Mauchly's W	Approx. Chi-Square	df	Sig.	Epsilon[b]		
					Greenhouse-Geisser	Huynh-Feldt	Lower-bound
Version	1.000	.000	0	.	1.000	1.000	1.000

Mauchly's test of sphericity indicates that the sphericity assumption is met for the repeated measures factor.

Tests of within-subjects effects

Measure: ProportionFound

Source		Type III Sum of Squares	df	Mean Square	F	Sig.
Version	Sphericity Assumed	.007	1	.007	.932	.345
	Greenhouse-Geisser	.007	1.000	.007	.932	.345
	Huynh-Feldt	.007	1.000	.007	.932	.345
	Lower-bound	.007	1.000	.007	.932	.345
Version * Group	Sphericity Assumed	3.841E-005	1	3.841E-005	.005	.942
	Greenhouse-Geisser	3.841E-005	1.000	3.841E-005	.005	.942
	Huynh-Feldt	3.841E-005	1.000	3.841E-005	.005	.942
	Lower-bound	3.841E-005	1.000	3.841E-005	.005	.942
Error(Version)	Sphericity Assumed	.154	22	.007		
	Greenhouse-Geisser	.154	22.000	.007		
	Huynh-Feldt	.154	22.000	.007		
	Lower-bound	.154	22.000	.007		

The omnibus F-test for the within-subjects effects is not statistically significant for the interaction effect nor the repeated measures factor.

Levene's test of equal variance

Measure: Proportion

Transformed Variable: Average

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Intercept	4.927	1	4.927	108.169	.000
Group	.864	1	.864	18.979	.000
Error	1.002	22	.046		

Levene's test shows homogeneity of variance for the between-groups factor, indicating that an F-test for the between-groups factor is appropriate.

Tests of between-subjects effects

Measure: ProportionFound

Transformed Variable: Average

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared	Noncent. Parameter	Observed Power[a]
Intercept	4.927	1	4.927	108.169	.000	.831	108.169	1.000
Group	.864	1	.864	18.979	.000	.463	18.979	.986
Error	1.002	22	.046					

An omnibus F-test for the between-subjects effect shows that the difference between the two groups is statistically significant and has a large effect size.

C.6.1 Miscellaneous descriptives

This section provides some miscellaneous descriptive statistics related to the experiment.

Mean proportion of errors found by group and paper version

	Group	Mean	Std. Deviation	N
ProportionFoundLow	Unprimed	.1754	.15832	12
	Primed	.4420	.17825	12
	Total	.3087	.21383	24
ProportionFoundHigh	Unprimed	.1969	.16453	12
	Primed	.4671	.14561	12
	Total	.3320	.20526	24

Pre-existing errors discovered in papers during experiment

Paper	Spell	Word	Article	VTP	P/P	Punc	Struc	Infor	Form	Total
AndroidFaces	2			1		1	2			6
HapticDesign	1			1	2		2			6
Notifications						1	1			2
Password	5						1			6

Additional errors within each paper discovered and classified after users in the study found them. These extra errors were included in the calculations for proportions of errors caught.

Average number of false positive errors per paper version

	AndroidFaces	HapticDesign	Notifications	Password
Low Error	32.0	3.0	14.5	5.7
High Error	5.2	27.5	10.8	6.5

Participants often made quite a few highlights that we determined were not actually errors.