Making the Web Easier to See with Opportunistic Accessibility Improvement

Jeffrey P. Bigham
Human-Computer Interaction and Language Technology Institutes
Carnegie Mellon University
jbigham@cs.cmu.edu

ABSTRACT
Many people would find the Web easier to use if content was a little bigger, even those who already find the Web possible to use now. This paper introduces the idea of *opportunistic accessibility improvement* in which improvements intended to make a web page easier to access, such as magnification, are automatically applied to the extent that they can be without causing negative side effects. We explore this idea with `oppaccess.js`, an easily-deployed system for magnifying web pages that iteratively increases magnification until it notices negative side effects, such as horizontal scrolling or overlapping text. We validate this approach by magnifying existing web pages $1.6x$ on average without introducing negative side effects. We believe this concept applies generally across a wide range of accessibility improvements designed to help people with diverse abilities.

Author Keywords
low vision; magnification; zoom; accessibility

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation: Misc.

INTRODUCTION
In real life, what someone can do is a function of not only a static notion of their abilities, but also their dynamic context and the effort needed to complete a task. This is particularly true of people with visual impairments accessing web content. Some use screen magnifiers to make computer content larger, others sit closer to the screen, and some do both (Figure 1). It is difficult for content creators to anticipate and adapt to the various display needs that individuals may have, and so often content is not laid out in a way that fully leverages the available space.

This is important because a large number of people could benefit from content that is a little bigger. While 39 million people worldwide are blind, many more (246 million) have low vision [6]. Even more people do not consider themselves to

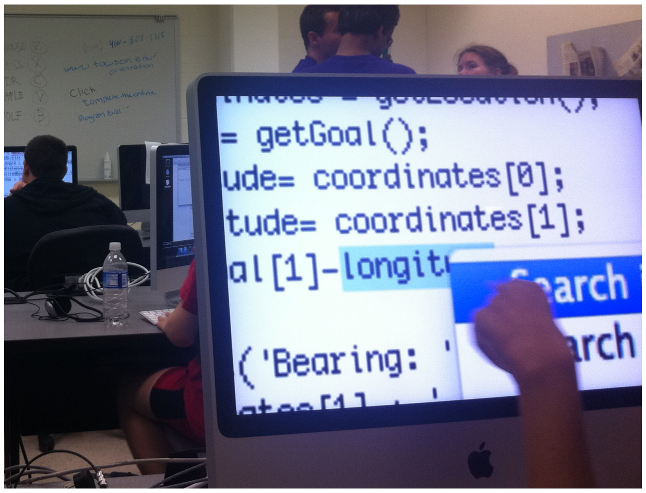


Figure 1. People with low vision use a combination of magnification software and physical accommodations like positioning themselves closer to the screen. *Opportunistic accessibility* is about improving accessibility when possible without adverse side effects, even if to a small degree.

have visual impairment and use no special software at all, but yet could benefit from improved accessibility, such as slightly larger content [3]. This is especially common among aging adults. Instead, these users may find their own solutions. For instance, they may prefer lower-resolution monitors on which content appears bigger, move their head closer to the screen when needed, or even prefer software programs or web sites that use larger fonts. Most are unaware that specialized software or features of their existing software exist, and many do not want (or would find it too complicated) to learn how to use it. As one example, we asked 50 Google Survey respondents over the age of 55 to tell us (without looking) how to make content in their web browser bigger. Only one wrote `ctrl+`, whereas the rest wrote some variant of "I don't know."

In this paper, we explore the potential of automatically making the Web display a little bit larger when we can do so without causing negative side effects. We call the generalization of this idea *opportunistic accessibility improvement*. In the case of web magnification, the idea is to make content a little bigger on web pages that a user visits, but only so much as can be done without introducing problems that otherwise affect the page. Opportunistic accessibility provides a new way of thinking about the sort of "messy," dynamic, and adjustable accessibility needs that people have. It may open a space for access technology that does not provide a perfect solution to problems, but rather makes access easier for people when it can, recognizing that they often find a way to access content even when the tool is unable to help.

We introduce `oppaccess.js`, a system that helps to make the Web a little easier to see. `oppaccess.js` does not overhaul the Web —in most cases we suspect users won't even notice the changes made unless told. Instead, content is just a little bigger so it is easier to see and a little easier to use. It detects common problems that occur when magnifying web content, and automatically sets its magnification level to avoid those problems.

We make three contributions: *(i)* we introduce the idea of *opportunistic accessibility*, *(ii)* we demonstrate the opportunity for making the Web a little bigger while minimizing negative impacts on usability, and *(iii)* we introduce `oppaccess.js` to make many web pages a little bigger when it can.

MAGNIFICATION AND THE WEB

`oppaccess.js` builds on prior work on magnification. ZoomText[1] is a popular example of the magnification software used by people with low vision. It graphically zooms screen content, which applies across a wide variety of content but tends to introduce horizontal scrolling (Figure 2a). Web pages are generally designed for vertical but not horizontal scrolling, and so horizontal scrolling can cause usability problems, *e.g.* needing to scroll constantly from left to right while reading a block of text. Another approach is to allow users to apply standard magnification in layers —for instance, applying some level of base magnification to the entire content area, and then a higher level of magnification to content of interest [5]. This is the approach taken by *sitecues*[2], which allows users to magnify web page content in layers.

Web browser magnification features have the potential to do better than graphical zoom. As content is magnified, the Web browser's layout engine reflows content to better utilize available space. The effect of magnification on web page layout is similar to what happens if the width of the window is decreased proportionally. For instance, a $2x$ zoom produces a layout similar to what would result if the window's width was divided by 2. The Web is characterized by diversity in both content and displays, and web browsers have developed sophisticated layout engines to reflow content to adapt. One simple example is wrapping of inline elements, such as text and images, within containing block-level elements.

Designing web pages that adapt to different devices is difficult, which has caused many web pages to be developed inflexibly assuming minimum requirements, *e.g.* for some time it was common to create web page content that was 800px wide and much of this content still exists. Internally, web browsers implement zoom by simply multiplying any fixed dimensions by the zooming factor. A DIV element that has been assigned a width 800px will be a fixed 1200px if the zoom level is set to $1.5x$. When web pages designed in this way are magnified they quickly grow beyond the width of the web browser window and introduce horizontal scrolling.

Responsive design allows web content to flexibly adapt to constraints. One component is CSS media queries, which allow different stylesheets to be applied based on different characteristics of the media the page is being displayed on, *e.g.* apply a stylesheet that displays less content horizontally when the width of the media is small. Another component is using flexibly-sized container elements. For instance, instead of setting the width of a DIV to 400px, its width might be set to 40%. The element then takes up 40% of the window, regardless of the magnification level that is applied.

Pages that use responsive design well can be magnified to higher levels of zoom before introducing problems like horizontal scrolling. At high magnification levels, flexibly-sized elements can result in two other problems. The first is that elements assigned percentage dimensions eventually start to overlap (Figure 2b), and the second is that growing text sizes with word wrapping can cause very narrow lines (Figure 2c).

`oppaccess.js` detect the three problems outlined above: *(i)* introduction of horizontal scrolling, *(ii)* overlapping DOM elements, and *(iii)* lines that are too narrow. It magnifies web content until it either detects one of these problems or reaches a maximum (currently $3x$).

More intelligent magnification of content has generally been more complicated and not yet been successfully ported to the Web. For instance, Supple automatically personalizes an interface to make it better for people with visual and motor disabilities [4], but it requires a formal description of the interface as input. `oppaccess.js` is not meant as a replacement for these, but as a much simpler alternative that may be "good enough" for many people. The fact that some of the more complicated approaches are not yet widely used may be seen as partial validation for our simpler *opportunistic accessibility* approach, even if it isn't as powerful.

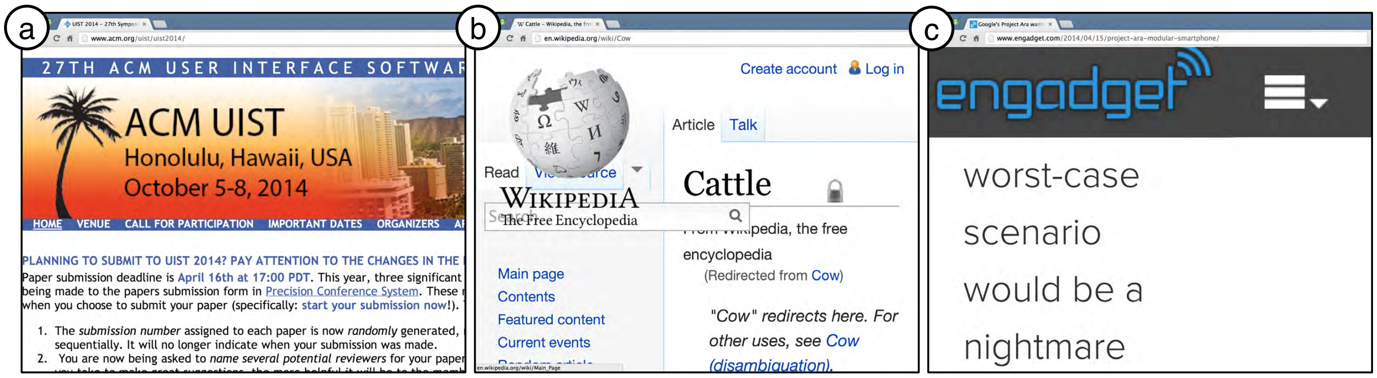
Past tools that do not fully replicate the functionality of existing solutions, yet are easy to access and transparent, have been successful. For instance, the WebAnywhere web-based screen reader has a subset of features available in screen readers and only works for the Web, but the fact that it can be accessed without installing new software has drawn hundreds of thousands of users [1]. `oppaccess.js` is similarly easy to install by injecting a single Javascript script via either a browser extension or by the web content developer.

MAKING THE WEB EASIER TO SEE

In this section, we describe `oppaccess.js`, our system for opportunistically magnifying web pages to make them easier to see. The system is implemented as a single cross-browser Javascript script (615 lines), not including jQuery. It can be delivered to web pages either by a web browser extension —we have implemented a Chrome extension —or by including the script directly in a web page. Thus, it can be used by either web developers to opportunistically improve the accessibility of their own web pages or by end users who want to apply it to all of the web pages they visit.

Magnifying web pages is surprisingly easy with Javascript. The base method is to programmatically set the value of the

[1]http://www.aisquared.com/zoomtext
[2]http://www.sitecues.com


a
b
c
UIST 2014 – 27th Sympos
www.acm.org/uist/uist2014/
W Cattle – Wikipedia, the free
en.wikipedia.org/wiki/Cow
Google's Project Ara wants
www.engadget.com/2014/04/15/project-ara-modular-smartphone/

27TH ACM USER INTERFACE SOFTWAR

ACM UIST
Honolulu, Hawaii, USA
October 5-8, 2014

HOME VENUE CALL FOR PARTICIPATION IMPORTANT DATES ORGANIZERS AI

PLANNING TO SUBMIT TO UIST 2014? PAY ATTENTION TO THE CHANGES IN THE
Paper submission deadline is April 16th at 17:00 PDT. This year, three significant
being made to the papers submission form in Precision Conference System. These
when you choose to submit your paper (specifically: start your submission now!).

1. The *submission number* assigned to each paper is now *randomly* generated,
sequentially. It will no longer indicate when your submission was made.
2. You are now being asked to *name several potential reviewers* for your paper

Create account Log in

Article Talk

Read View source

WIKIPEDIA
The Free Encyclopedia

Cattle

From Wikipedia, the free encyclopedia

(Redirected from Cow)

"Cow" redirects here. For other uses, see Cow (disambiguation).

Main page
Contents
Featured content
Current events

engadget

worst-case
scenario
would be a
nightmare

Figure 2. **Illustrated examples of three of the most common problems introduced by web magnification: *(a)* graphical zoom introduces horizontal scrolling, *(b)* responsive design works well to adapt a web page to higher levels of magnification but often results in overlapping content as magnification grows too large, and *(c)* as the size of text grows too large relative to its container the lines of text become undesirably thin.**

CSS property `document.body.style.zoom`. For instance, setting this property to 2.0 results in a doubling of the size of page elements. The effect is quite similar to what results when a user presses `ctrl +` or `cmd +` several times to manually magnify the page. Using the script, the magnification is optimally applied as soon as the page is loaded. The script uses a page-specific cookie to remember the optimal zoom level for the given web page, so that it can be magnified immediately next time.

Page Health

To determine the magnification level to apply, `oppaccess.js` iteratively increases magnification in steps of 0.1 and checks after each to see if it decreases what we call *page health*. The page health is currently defined as 1 minus the sum of the area of the page that *(i)* is horizontally scrolled out of the viewport, *(ii)* overlaps with another element, or *(iii)* contains narrow word-wrapped text. The area considers the full dimensions of the page, regardless of whether or not they are visible. The page health is calculated and recorded when the page loads because some pages exhibit these problems when they are first loaded. Iterative magnification is stopped whenever the page health decreases by more than 0.05. Modern web browsers are smart enough not to visually re-render the page in between iterations, and so it appears to happen all at once.

Horizontal Scrolling

Horizontal scrolling occurs when the width of the web page content is larger than the width of the viewport. Horizontal scrolling can be detected by comparing the width of a web page's `document` element to the width of the `window`. This is efficient and robust to check, and we have found it to be the first problem introduced when magnifying most web pages.

Overlapping Text

When elements are assigned CSS positioning other than the default, e.g. static, absolute, relative, there is potential for elements to overlap. The likelihood of this happening increases when web pages are magnified because usually the web page author has not carefully thought about or designed for this case. To detect area of the page that has overlapping text areas, `oppaccess.js` first finds all elements that contain text, records their location and dimension, and then runs an efficient line-scanning algorithm (also implemented in Javascript) to find overlapping area. It then returns the sum of the overlapped area.

Narrow Word Wrapping

A common problem encountered on web pages that use container elements with percentage widths is that as the text they contain grows larger with magnification, the number of words per line decreases (Figure 2c). `oppaccess.js` detects this sort of problem with word wrapping by checking each block-level element that contains text to ensure that enough words are able to be displayed per line (5 is currently used as the cut-off). It ignores elements that do not contain enough words ($2x$ the cut-off, so 10 in this example).

VALIDATING OUR APPROACH

In order to understand the diversity of the web in terms of what might be difficult for a person with low vision to see, we conducted a study with 50 Mechanical Turk workers. Each worker was instructed to enter the URL of a web page that they either currently had open in their browser or that they had recently visited (other than mturk.com). We also captured the width and height of each worker's window. The average window width was 1236px (SD=232.9), which was our primary concern since height does not influence page health. We then ran `oppaccess.js` on the URLs provided by each worker in a window with the width and height set to what the worker had provided. We recorded the magnification that `oppaccess.js` achieved, which component of page health prevented it from magnifying the page further, and the time required to do the calculation.

Overall, the web pages were magnified 1.6x (SD=0.9) without causing a significant decrease in page health (Figure 3). Magnification stopped on 41 of the web pages because of horizontal scrolling, on 6 because of narrow word wrapping, and on only 2 because of text overlap (Table 1). We also computed the achieved magnification level for web browsers with pixels widths from 500 to 2000. As expected, `oppaccess.js` is able to increase the magnification more as available width increases (Figure 4). Figure 5 shows the effect of `oppaccess.js` on the UIST 2014 homepage.

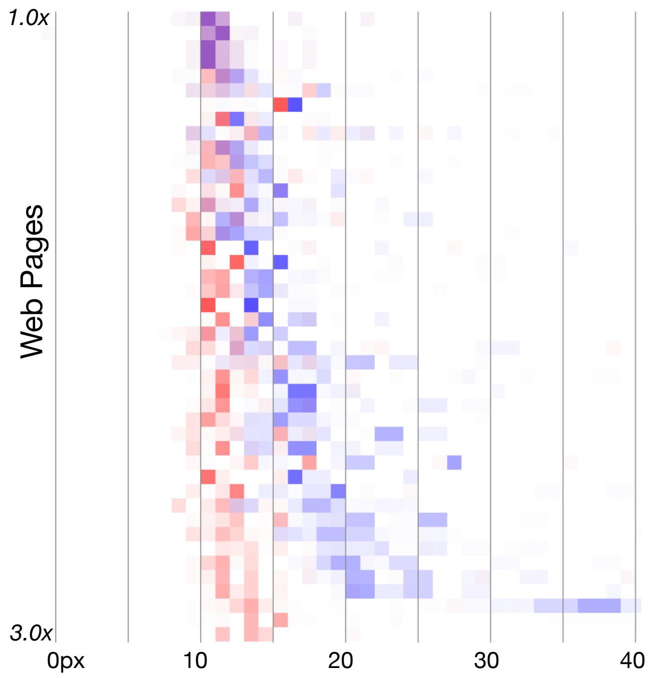


Figure 3. Font size distribution across 50 web pages (red) and the new spread when applying `oppaccess.js` (blue). Each row is a web page, and each column is a font size. Web pages are ordered by the level of magnification applied. The effect is to pull the font sizes higher.

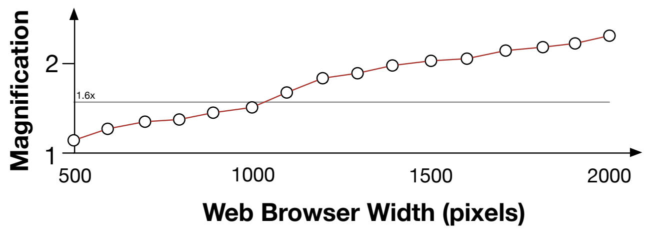
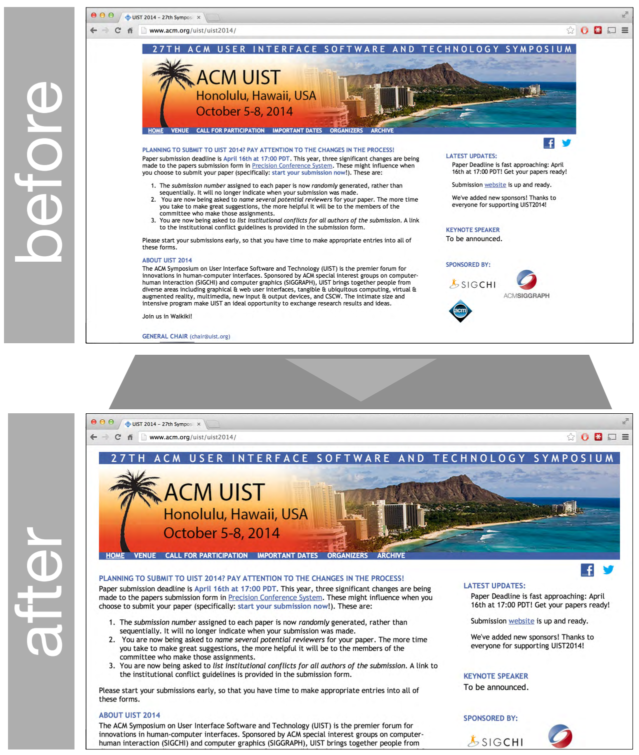


Figure 4. Achieved magnification level across different web browser widths. `oppaccess.js` is able to magnify content more when the window size is larger.

The time required to compute the magnification was 2.2 seconds (SD=3) on average, although this was dominated by the time required to check text overlap and word wrapping, *e.g.* those elements that result from responsive design. While web pages are likely to include these even more frequently in the future, we believe the performance of our implementation could be dramatically improved by either building it into the browser, using predictive methods, or by sharing the magnification levels that prior visitors with similar window sizes found useful. We conservatively increased magnification in an iterative fashion in case the browser rendered intermediate steps; in the future, a binary search approach may be used because modern browsers do not render the intermediate steps.

DISCUSSION & FUTURE WORK

This paper has introduced the idea of opportunistic accessibility improvement and validated the concept for web magnification by showing that `oppaccess.js` can magnify content by 60% without decreasing page health. We do not claim to have developed a robust solution to web access for people with visual impairments, only to have introduced a tool that will make web content slightly larger, which will benefit people who currently find it difficult to see. Future work may seek to better understand how much of a difference small improvements make.



PLANNING TO SUBMIT TO UIST 2014? PAY ATTENTION TO THE CHANGES IN THE PROCESS!

Paper submission deadline is April 16th at 17:00 PDT. This year, three significant changes are being made to the papers submission form in Precision Conference System. These might influence when you choose to submit your paper (specifically: start your submission now!). These are:

1. The *submission number* assigned to each paper is now *randomly* generated, rather than sequentially. It will no longer indicate when your submission was made.
2. You are now being asked to *name several potential reviewers* for your paper. The more time you take to make great suggestions, the more helpful it will be to the members of the committee who make those assignments.
3. You are now being asked to *list institutional conflicts for all authors of the submission*. A link to the institutional conflict guidelines is provided in the submission form.

Please start your submissions early, so that you have time to make appropriate entries into all of these forms.

ABOUT UIST 2014

The ACM Symposium on User Interface Software and Technology (UIST) is the premier forum for innovations in human-computer interfaces. Sponsored by ACM special interest groups on computer-human interaction (SIGCHI) and computer graphics (SIGGRAPH), UIST brings together people from diverse areas including graphical & web user interfaces, tangible & ubiquitous computing, virtual & augmented reality, multimedia, new input & output devices, and CSCW. The intimate size and intensive program make UIST an ideal opportunity to exchange research results and ideas.

Join us in Waikiki!

GENERAL CHAIR (chair@uist.org)

LATEST UPDATES:

Paper Deadline is fast approaching: April 16th at 17:00 PDT! Get your papers ready!

Submission website is up and ready.

We've added new sponsors! Thanks to everyone for supporting UIST2014!

KEYNOTE SPEAKER

To be announced.

SPONSORED BY:

Figure 5. The observed result of applying *oppaccess.js* to the UIST 2014 web page. In the *after* screenshot the page is automatically magnified to 1.3x the original, while ensuring that negative effects (horizontal scrolling, overlapping elements, or narrow word-wrapping) are not introduced. This may seem like a small change, but can improve the realized accessibility of this web page for many people with low vision.

Future work may also extend our approach for magnifying the Web. yIn particular, it may be possible to allow the user more direct control of how much magnification they would like to have. In the case that the desired magnification level has not been reached, we may involve the user in deciding how and to what extent they may be comfortable in reducing page health. Education about existing browser features, such as `ctrl +` are likely to be part of the solution. We might substitute other magnification strategies likely to affect the page in different ways or applied only to portions of the page that are not yet experiencing problems.

While the page health measures for text overlap and narrow word wrapping affected only 14% of web pages, it seems likely that this proportion will increase as responsive design becomes more popular. Elements of our methods for detecting when responsive design elements break down may find broader application. While the majority of the population may not use high levels of magnification, the duality between screen size and magnification means that these could be relevant for controlling responsiveness on small screen devices.

Many areas within accessibility could benefit from the idea of opportunistic accessibility improvement, especially those areas characterized by enormous variability. Problems for which people are currently "getting by" and could benefit from even small improvements seem most promising, such as those resulting from visual, motor, or cognitive impairments. The concept of opportunistic accessibility may open up the possibilty for tools to make targets a little easier to click or web interfaces a bit easier to understand.

To some extent, accessibility is already opportunistic. Assistive technology often only works correctly when content and software is produced in a way that allows it to work, which they often are not. As another example, the WebInSight system provided labels for images so that blind people could access them, but could provide labels for less than half of them [2]. Nevertheless, this technology is useful because of the increased access it provides when it does work, and so in some sense the idea is not new. Articulating the opportunistic nature of these tools may lead future work to embrace and leverage our inability to always make access perfect.

CONCLUSION

We have introduced *opportunistic accessibility improvement* and explored it through `oppaccess.js`, an easily-deployed Javascript system for magnifying web pages. We have shown that the Web could be on average 60% larger for people on the Web without introducing the most common negative effects of magnification. The approach of improving accessibility when possible without introducing negative side effects may make seemingly small improvements, but if broadly applied could improve experienced accessibility for a wide range of people.

ACKNOWLEDGEMENTS

This paper was developed under a grant from the Department of Education, NIDRR grant number H133A130057, National Science Foundation Award #IIS-1149709, and an Alfred P. Sloan Foundation Fellowship. We thank our anonymous reviewers for their many helpful comments.

REFERENCES

1. Bigham, J.P., Prince, C.M., and Ladner, R.E. WebAnywhere: A Screen Reader On-the-Go. W4A 2008. 73-82.
2. Bigham, J.P., Kaminsky, R.S., Ladner, R.E., Danielsson, O.M., and Hempton, G.L. WebInSight: making web images accessible. ASSETS 2006. 181-188.
3. Forrester Research. Accessible Technology in Computing —Examining Awareness, Use, and Future Potential. 2004.
4. Gajos, K.Z., Wobbrock, J.O., and Weld, D.S. Automatically generating user interfaces adapted to users' motor and vision capabilities. UIST 2007. 231-240.
5. Kline, R.L. and Glinert, E.P. Improving GUI accessibility for people with low vision. CHI 1995. 114-121.
6. World Health Organization. Visual impairment and blindness. 2013. http://www.who.int/mediacentre/factsheets/fs282/en/

URL	Magnification	Stopping Cause
http://www.espncricinfo.com/	1	scroll
http://www.oemcycle.com/Item/product/900076028/_Vid89455142n47	1	scroll
http://www.espncricinfo.com/	1	scroll
http://www.espncricinfo.com/ci/engine/current/match/scores/live.html	1	scroll
https://www.facebook.com/profile.php?id=XXXXXXXXXXX	1.1	scroll
http://www.imdb.com/title/tt1843866/reviews?ref=tt_urv	1.1	scroll
http://ibnlive.in.com/	1.1	scroll
http://bleacherreport.com/	1.1	scroll
http://torrage.com/	1.1	scroll
http://nav.sbilife.co.in/UI/DailyNavInputOutput.aspx	1.1	scroll
http://espn.go.com/nba/draft2014/story/_/id/10780763/	1.1	wrap
http://www.scribd.com/doc/61445642/Boolean-Expressions	1.2	scroll
http://www.goodbelly.com	1.2	scroll
https://www.facebook.com/fayiso77	1.2	scroll
http://hip2save.com	1.3	scroll
https://www.facebook.com/	1.3	scroll
http://www.hcrealms.com/index.php?page=units	1.3	scroll
http://radar.weather.gov/ridge/Conus/centgrtlakes_loop.php	1.3	scroll
https://www.mturk.com/mturk/findhits?match=true	1.3	scroll
http://www.limeroad.com/	1.3	scroll
http://www.pasadrexam2014.in/	1.3	scroll
http://oldnavy.gap.com/browse/category.do?cid=6825	1.3	scroll
https://www.google.com/	1.3	scroll
http://www.youtube.com/watch?v=jkNyC7FPJxc	1.3	scroll
http://slickdeals.net/	1.3	text
http://hw.reddit.com/r/sixwordstories/	1.4	scroll
http://www.mturkgrind.com/threads/15270-04-14-Happy-Black-Day-Monday!/page113	1.4	scroll
http://en.wikipedia.org/wiki/Tariq_Farid	1.4	wrap
http://translate.eu/	1.5	scroll
http://www.reddit.com/	1.5	scroll
https://tools.usps.com/go/ZipLookupAction_input	1.5	scroll
http://money.cnn.com/?hpt=sitenav	1.5	scroll
https://accounts.google.com/ServiceLogin?service=mail	1.5	scroll
http://en.wikipedia.org/wiki/Alexander_Mirtchev	1.5	wrap
http://www.noxxic.com/wow/pve/warlock/demonology/talent-build-and-glyphs	1.6	scroll
http://www.inboxdollars.com/search/...	1.6	scroll
https://hr.macys.net/iip/Login.aspx	1.6	scroll
http://www.oddee.com/	1.6	scroll
https://mail.google.com/mail/u/0/#inbox	1.7	scroll
https://www.cedarpoint.com/blog-article/online-fun/Winter-Scenes	1.7	scroll
http://foodomania.com/mor-kuzhambu/	1.7	scroll
https://us.battle.net/support/en/games/diablo3	1.7	scroll
https://chrome.google.com/webstore/category/apps?hl=en	1.8	wrap
http://www.twitch.tv/manvsgame	1.9	text
http://en.wikipedia.org/wiki/Italian_art	1.9	wrap
http://timesofindia.indiatimes.com/international-home	2.1	scroll
http://www.wral.com/severe-weather-freezing-temps-expected-tuesday/13567824/	2.8	scroll
http://www.pinterest.com/	3	none
https://www.flickr.com/photos/elisa_maza/12942331964/in/...	3	scroll
http://www.msnbc.com/rachel-maddow-show/the-bundy-crisis-nevada	3	wrap

Table 1. The web page URLs gathered during our Mechanical Turk study, ordered by the magnification `oppaccess.js` achieved on each and what caused magnification to stop —horizontal scrolling, text overlap, or narrow word wrapping.