Inverse Color Contrast Checker: Automatically Suggesting Color Adjustments that meet Contrast Requirements on the Web

Frode Eika Sandnes

Oslo Metropolitan University, frodes@oslomet.no

Low contrast between text and background and its effect with low vision is relatively well-understood. Many tools exist for helping web designers check contrast limits. Most of these tools identify contrast problems but give limited advice on how to rectify the problems. Moreover, website accessibility audits reveal that insufficient color contrast still is a recurring issue in practice. A framework was therefore developed that automatically proposes color adjustments to problematic text-background color pairs on web pages. These suggestions adhere to contrast requirements and are aligned with the visual design profile. The framework allows the developers to visually inspect the suggestions and amend color definitions in projects.

CCS CONCEPTS • Human-centered computing~Accessibility

Additional Keywords and Phrases: web accessibility, color contrast, WCAG, web framework

ACM Reference Format:

First Author's Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

1 Introduction

Human visual perception is based on detecting differences. Without differences no visual information can be perceived. To visually perceive text there must be sufficient contrast between the text and its background. The level of contrast needed depends on the size of visual elements as smaller elements require a higher level of contrast than larger elements. Consequently, individuals with reduced visual acuity require the contrast level to be higher than individuals with uncorrected vision.

The Web Content Accessibility Guidelines (WCAG2.1) gives very concrete advice about the minimum contrast levels that should be used for headings and body text on the web [28]. Many tools exist for checking the contrast levels of text-background color pairs [1, 9]. Yet, recent accessibility audits of public websites reveal that insufficient contrast is still a common problem [2]. Often developers must adhere to given graphical design profiles which may take precedence over accessibility guidelines. Although websites may appear visually coherent, it is our observation that many websites are ineffectively coded with inconsistent color definitions scattered in different locations throughout the code increasing the chance that contrast issues pass unnoticed.

The goal of this work was to give developers an alternative view of color contrast within a website under development. The tool identifies all instances of text with insufficient contrast and visually shows how these issues could be corrected with the smallest changes possible while preserving the graphical design profile. If opting for the maximum contrast level possible (black and white) the color profile will be lost.

2 Related work

There has been much research into color and readability. Most notably color choices can affect access for individuals with reduced perception to color [4, 15] and reduced visual acuity [7, 10]. Color can also affect access for individuals with dyslexia [17], situational visual impairments [24, 25], the degree of fatigue during prolonged use [31], or the perceived user experience [14].

Several contrast checking tools allow designers to manually check the luminance contrast of color pairs while tools such as WAVE and Button Contrast Checker automatically check entire web pages [1, 9]. Simulators have also been proposed as a means of drawing attention towards accessibility issues [9]. However, it has been argued that one should not rely on simulators alone and that these may amplify disability stereotypes [26].

Color choice decisions may be taken while using design tools, and it has been argued that design tools should promote accessibility [30]. Several studies have explored color picking interfaces [3, 4, 5, 6] and alternative color picking tools have been proposed [11, 12]. The visual nature of visual tools allows contrast information to be integrated into the color picking interface [16, 23] to increase designer and developers understanding of contrast limitations [20, 21], for example in the RGB [18] and HSV [19] color spaces, or palettes that compensates for reduced color vision [27]. Similar ideas have also been incorporated into mainstream tools such as DevTools bundled with the Chrome browser. It has been found that contrast tools are effective in detecting contrast issues when the text background is solid, but these tools have limitations if the text appears on top of images, or text appears within images [1].

Methods for adjusting colors for reduced color vision have been proposed [29] where the full color space is transformed to a reduced color space matching the type of color vision. The general rule-of-thumb for fixing contrast issues in general is to increase the difference in brightness. This rule-of-thumb has been proposed to suggest contrast adjustments using pixel-level analysis of visual material [22] and a web-tool for getting pixel-based color adjustments was proposed [8]. The framework proposed herein makes suggestions based on the same principle but instead of pixel level analysis, the framework modifies the DOM-structure of web pages directly. A totally different approach is to communicate color information non-visually using haptics [13].

3 The Inverse Color Contrast Checker Framework

The Inverse Color Contrast Checker Framework comprises a JavaScript library that is attached to an existing web project, typically by setting up a callback through the onload event of the body tag or by inserting a button on the page that activates the framework (modern browsers do not allow cross site scripting). The developer therefore needs to be able to edit the html source. The tool can also be used with third-party websites by downloading a local copy. When the connected web page is loaded in the browser the tool is controlled by a series of keystrokes to cycle between the original color settings, color settings with contrast levels adhering to WCAG2.1 Level AA and WCAG2.1 Level AAA (see Figures 1, 2 and 3). Each color definition is handled individually. Visual (togglable) markers show the corrected elements. There are also keystrokes to adjust the saturation level and brightness level of all text of the web page simultaneously. Details about the suggested corrections are logged in the browser console window (see Figure 4). These details can in turn be used to update the color definitions for a given project.

The framework traverses the Document Object Model (DOM) of the web page for all color definitions. The contrast level of each color pair instance is checked and adjusted if it does not adhere to the current requirement. Adjustment suggestions are determined by first numerically solving the luminance contrast inequation for a text brightness level that gives sufficient contrast, failing that, the inequation is numerically solved for the text-color saturation level that gives sufficient contrast. If there are no suitable brightness and saturation levels for the text, the brightness and saturation level of the background are also altered in a similar manner until a suitable set of adjustments are found. The inequalities are solved using linear search (each of brightness and saturation from 0 to 100% in steps of 1%). Note that the computed color of text and the inherited background color were used as the rendered color of an element may be affected by other elements.

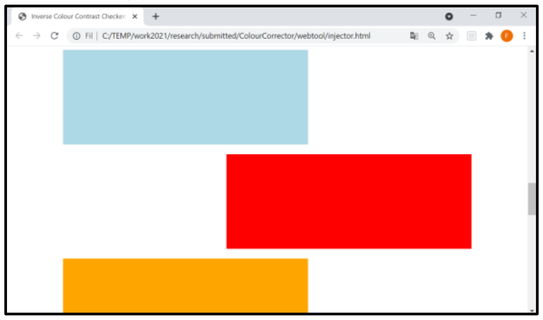
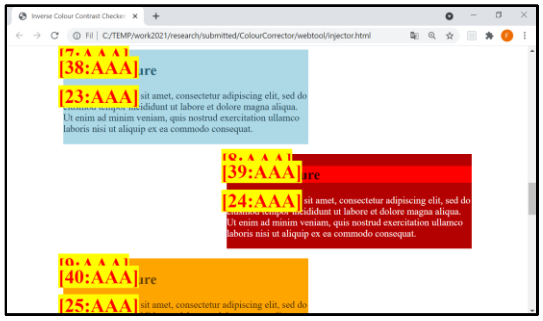
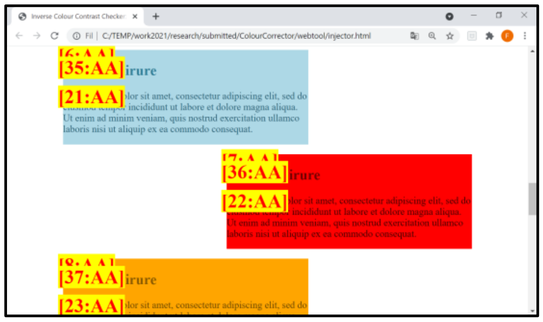

Inverse Colour Contrast Checker
C:/TEMP/work2021/research/submitted/ColourCorrector/webtool/injector.html

Figure 1. Original web page sample with identical text and background colors (no contrast).


Inverse Colour Contrast Checker
C:/TEMP/work2021/research/submitted/ColourCorrector/webtool/injector.html
[38:AAA]
[23:AAA]
[39:AAA]
[24:AAA]
[40:AAA]
[25:AAA]

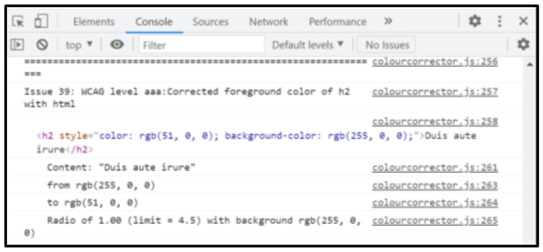
sit amet, consectetur adipiscing elit, sed do
incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris nisi ut aliquip ex ea commodo consequat.

Figure 2. Sample web page automatically corrected for WCAG Level AAA (optional) contrast recommendations.


Inverse Colour Contrast Checker
C:/TEMP/work2021/research/submitted/ColourCorrector/webtool/injector.html
[35:AA] irure
[21:AA]
[36:AA] irure
[22:AA]
[37:AA] irure
[23:AA]

lor sit amet, consectetur adipiscing elit, sed do
incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation ullamco
laboris nisi ut aliquip ex ea commodo consequat.

Figure 3. Sample web page automatically corrected for WCAG Level AA contrast recommendations.



```
============================================================ colourcorrector.js:256
===
Issue 39: WCAG level aaa:Corrected foreground color of h2    colourcorrector.js:257
with html
                                                             colourcorrector.js:258
  <h2 style="color: rgb(51, 0, 0); background-color: rgb(255, 0, 0);">Duis aute
  irure</h2>
    Content: "Duis aute irure"                               colourcorrector.js:261
    from rgb(255, 0, 0)                                      colourcorrector.js:263
    to rgb(51, 0, 0)                                         colourcorrector.js:264
    Radio of 1.00 (limit = 4.5) with background rgb(255, 0,  colourcorrector.js:265
0)
```

Figure 4. Color details in the browser console window.

4 Experiences

The framework was tested on the main page of four public websites (conference, education, newspaper, and government agency). Local copies of the pages were downloaded using "save as" in Google Chrome and onload callbacks to the framework were added to the body tags. The results were visually inspected. The results were checked against WAVE.

The framework suggested Level AA and AAA contrast corrections on three of the four webpages. The Level AA suggestions on two webpages were triggered by definitions with no visual implications (false positives). WAVE confirmed the Level AA (recommended) contrast errors for one of the webpages. The visual markers were helpful in drawing attention towards the more subtle changes that were otherwise hard to spot manually. The suggested changes appeared to be consistent with the respective color profiles. The tests revealed a common practice for using the !mportant CSS directive which made the rendered elements immune to the framework. The code in most of the downloaded pages seemed overly verbose.

One challenge with some websites is that local page download in the browser is incomplete since the pages are delivered by some content management systems and, in some cases, simple manual intervention was needed to deactivate JavaScript that caused problems. Contrast issues in nested html-structures are reported multiple times (at each level). Moreover, the framework does not consider the net effect of transparency and image backgrounds.

5 Conclusions

A framework for working with color contrast during web development was presented. Developers can visualize the automatically suggested color corrections that satisfy WCAG2.1 contrast requirements and adhere to the original design profile. The suggestions may help visualize opportunities for meeting Level AAA conformance. The framework needs to be manually attached to webpages. However, this should not be a hindrance as developers usually have access to, control over, and understanding of their own code. The framework is available at https://github.com/frode-sandnes/INJECTOR and a live demo can be run at https://www.cs.oslomet.no/~frodes/INJECTOR/.

REFERENCES

<bib id="bib1"><number>[1]</number>Rafael Almeida and Carlos Duarte. 2020. Analysis of automated contrast checking tools. In Proceedings of the 17th International Web for All Conference (W4A '20). Association for Computing Machinery, New York, NY, USA, Article 18, 1–4. DOI:https://doi.org/10.1145/3371300.3383348</bib>
<bib id="bib2"><number>[2]</number>Agency for Universal Design of ICT (uu-tilsynet). 2021. Accessibility audit report depository. Retrieved June 16, 2021 from https://www.uutilsynet.no/tilsynsrapporter/tilsynsrapporter/270 </bib>
<bib id="bib3"><number>[3]</number>Kristian Brathovde, Mads Brændeland Farner, Fredrik Krag Brun, and Frode Eika Sandnes. 2019. Effectiveness of Color-Picking Interfaces among NonDesigners, In Proceedings of the 16th International Conference on Cooperative Design, Visualization and Engineering 2019 (CDVE 2019), LNCS vol. 11792, Springer, pp. 181-189. DOI: https://doi.org/10.1007/978-3-030-30949-7_21 </bib>
<bib id="bib4"><number>[4]</number>Ricardo José de Araújo, Julio Cesar Dos Reis, and Rodrigo Bonacin 2020. Understanding interface recoloring aspects by colorblind people: a user study. Universal Access in the Information Society 19.1 (2020): 81-98.</bib>
<bib id="bib5"><number>[5]</number>Sarah A. Douglas and Arthur E. Kirkpatrick. 1999. Model and representation: the effect of visual feedback on human performance in a color picker interface. ACM Trans. Graph. 18, 2 (April 1999), 96-127. DOI: http://dx.doi.org/10.1145/318009.318011 </bib>
<bib id="bib6"><number>[6]</number>Berto Gonzalez and Celine Latulipe. 2011. BiCEP: bimanual color exploration plugin. In CHI '11 Extended Abstracts on Human Factors in Computing Systems (CHI EA '11). ACM, New York, NY, USA, 1483-1488. DOI: https://doi.org/10.1145/1979742.1979795 </bib>

<bib id="bib7"><number>[7]</number>Massimo Greco, Natale Stucchi, Daniele Zavagno, and Barbara Marino. 2008. On the portability of computer-generated presentations: The effect of text-background color combinations on text legibility. Human factors 50, 5 (2008), 821-833. DOI: https://doi.org/10.1518%2F001872008X354156</bib>
<bib id="bib8"><number>[8]</number>Fredrik Hansen, Josef Jan Krivan, and Frode Eika Sandnes. 2019. Still Not Readable? An Interactive Tool for Recommending Color Pairs with Sufficient Contrast based on Existing Visual Designs. In The 21st International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '19). Association for Computing Machinery, New York, NY, USA, 636–638. DOI:https://doi.org/10.1145/3308561.3354585</bib>
<bib id="bib9"><number>[9]</number>Shashank Kumar, JeevithaShree DV, and Pradipta Biswas. 2020. Accessibility evaluation of websites using WCAG tools and Cambridge Simulator. arXiv preprint arXiv:2009.06526.</bib>
<bib id="bib10"><number>[10]</number>M. L. Mathews. 1989. Visual performance with coloured CRT displays: research update. Appl. Ergon. 20, 58 (1989), 58. https://doi.org/10.1016/0003-6870(89)90011-2</bib>
<bib id="bib11"><number>[11]</number>Barbara J. Meier, Anne Morgan Spalter, and David B. Karelitz. 2004. Interactive color palette tools. IEEE Computer Graphics and Applications 24, 3 (2004), 64- 72. DOI: https://doi.org/10.1109/MCG.2004.1297012 </bib>
<bib id="bib12"><number>[12]</number>Giovanni Moretti and Paul Lyons. 2002. Tools for the selection of colour palettes. In Proceedings of the SIGCHI-NZ Symposium on Computer-Human Interaction (CHINZ '02). ACM, New York, NY, USA, 13-18. DOI: http://dx.doi.org/10.1145/2181216.2181219 </bib>
<bib id="bib13"><number>[13]</number>Richard Nguyen and Connor Geddes. 2019. Exploring Haptic Colour Identification Aids. In The 21st International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '19). Association for Computing Machinery, New York, NY, USA, 709–711. DOI:https://doi.org/10.1145/3308561.3356111</bib>
<bib id="bib14"><number>[14]</number>Lasse Apalnes Pedersen, Svavar Skuli Einarsson, Fredrik Arne Rikheim, and Frode Eika Sandnes. 2020. User Interfaces in Dark Mode During Daytime–Improved Productivity or Just Cool-Looking?. In International Conference on Human-Computer Interaction. Springer, Cham. 178-187. DOI: https://doi.org/10.1007/978-3-030-49282-3_13</bib>
<bib id="bib15"><number>[15]</number>Gajo Petrovic and Hamido Fujita. 2017. Deep Correct: Deep Learning Color Correction for Color Blindness. SoMeT, pp. 824-834.</bib>
<bib id="bib16"><number>[16]</number>Katharina Reinecke, David R. Flatla, and Christopher Brooks. 2016. Enabling Designers to Foresee Which Colors Users Cannot See. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). ACM, New York, NY, USA, 2693- 2704. DOI: https://doi.org/10.1145/2858036.2858077.</bib>
<bib id="bib17"><number>[17]</number>Luz Rello and Jeffrey P. Bigham. 2017. Good Background Colors for Readers: A Study of People with and without Dyslexia. In Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '17). Association for Computing Machinery, New York, NY, USA, 72–80. DOI:https://doi.org/10.1145/3132525.3132546</bib>
<bib id="bib18"><number>[18]</number>Frode Eika Sandnes and Anqi Zhao. 2015. An interactive color picker that ensures WCAG2. 0 compliant color contrast levels. Procedia Computer Science 67 (2015), 87-94. DOI: https://doi.org/10.1016/j.procs.2015.09.252 </bib>
<bib id="bib19"><number>[19]</number>Frode Eika Sandnes and Anqi Zhao. 2015. A contrast colour selection scheme for WCAG2.0-compliant web designs based on HSV-half-planes. In Proceedings of the 2015 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2015). IEEE, 1233-1237. DOI: https://doi.org/10.1109/SMC.2015.220 </bib>
<bib id="bib20"><number>[20]</number>Frode Eika Sandnes. 2016. Understanding WCAG2.0 color contrast requirements through 3D color space visualization. Studies in health technology and informatics 229 (2016), 366-375. DOI: https://doi.org/10.3233/978-1-61499-684-2-366 </bib>
<bib id="bib21"><number>[21]</number>Frode Eika Sandnes. 2017. On-screen colour contrast for visually impaired readers. In: Information Design: Research and Practice. Alison Black, Paul Luna, Ole Lund and Sue Walker (eds.). Routledge, 405-416.</bib>
<bib id="bib22"><number>[22]</number>Frode Eika Sandnes. 2018. An image-based visual strategy for working with color contrasts during design. In Proceedings of the International Conference on Computers Helping People with Special Needs (ICCHP), LNCS Vol. 10896, Springer, Cham, 35-42. DOI: https://doi.org/10.1007/978-3-319-94277-3_7 </bib>
<bib id="bib23"><number>[23]</number>Garreth W. Tigwell, David R. Flatla, and Neil D. Archibald. 2017. ACE: A Colour Palette Design Tool for Balancing Aesthetics and Accessibility. ACM Trans. Access. Comput. 9, 2, Article 5 (January 2017), 32 pages. DOI: https://doi.org/10.1145/3014588.</bib>
<bib id="bib24"><number>[24]</number>Garreth W. Tigwell, David R. Flatla, and Rachel Menzies. 2018. It's not just the light: understanding the factors causing situational visual impairments during mobile interaction. In Proceedings of the 10th Nordic Conference on Human-Computer Interaction (NordiCHI '18). Association for Computing Machinery, New York, NY, USA, 338–351. DOI:https://doi.org/10.1145/3240167.3240207</bib>
<bib id="bib25"><number>[25]</number>Garreth W. Tigwell, Rachel Menzies, and David R. Flatla. 2018. Designing for Situational Visual Impairments: Supporting Early-Career Designers of Mobile Content. In Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18). Association for Computing Machinery, New York, NY, USA, 387–399. DOI:https://doi.org/10.1145/3196709.3196760</bib>
<bib id="bib26"><number>[26]</number>Garreth W. Tigwell. 2021. Nuanced Perspectives Toward Disability Simulations from Digital Designers, Blind, Low Vision, and Color Blind People. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21). Association for Computing Machinery, New York, NY, USA, Article 378, 1–15. DOI:https://doi.org/10.1145/3411764.3445620</bib>
<bib id="bib27"><number>[27]</number>Luigi Troiano, Cosimo Birtolo, and Maria Miranda. 2008. Adapting palettes to color vision deficiencies by genetic algorithm. In Proceedings of the 10th annual conference on Genetic and evolutionary computation (GECCO '08), Maarten Keijzer (Ed.). ACM, New York, NY, USA, 1065-1072. DOI: https://doi.org/10.1145/1389095.1389291 </bib>
<bib id="bib28"><number>[28]</number>W3C, WCAG2.1. Retrieved June 16, 2021 from https://www.w3.org/TR/WCAG21/ </bib>
<bib id="bib29"><number>[29]</number>Ken Wakita and Kenta Shimamura. 2005. SmartColor: disambiguation framework for the colorblind. In Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility (Assets '05). Association for Computing Machinery, New York, NY, USA, 158–165. DOI:https://doi.org/10.1145/1090785.1090815</bib>
<bib id="bib30"><number>[30]</number>Montgomery Webster. 2014. Integrating color usability components into design tools. interactions 21, 3 (May 2014), 56-61. DOI: https://doi.org/10.1145/2591512</bib>
<bib id="bib31"><number>[31]</number>Xiaojiao Xie, Fanghao Song, Yan Liu, Shurui Wang, and Dong. 2021. Study on the Effects of Display Color Mode and Luminance Contrast on Visual Fatigue. IEEE Access, 9, 35915-35923. DOI: https://doi.org/10.1109/ACCESS.2021.3061770</bib>