Autonomous Agents for Accessibility: Simulating Visual Impairments in Web Interfaces

Juan Diego Yepes-Parra ⓘD, Camilo Escobar-Velásquez ⓘD
Universidad de los Andes, Colombia
{j.yepes, ca.escobar2434} @uniandes.edu.co

Abstract—Static code analysis cannot detect real-time interaction issues faced by users with disabilities. We propose a multi-modal Artificial Intelligence (AI) agent framework that simulates interactions of users with visual impairments without code access. The agent would closely simulate the experience of these users by interacting with web interfaces using the same modalities available to them, primarily keyboard navigation and screen readers. The agent perceives the interface through perceptual filters that mimic conditions including glaucoma and myopia, and handles both the altered visual input and audio output from screen readers. This approach aims to replicate the real-world constraints and strategies of users with disabilities, enabling more realistic evaluation, with the objective of identifying, locating and repairing web accessibility issues. We suggest a framework to evaluate how such filters affect user behavior, task success, and User Interface (UI) usability. Our approach aims to uncover visual accessibility flaws that become apparent under impaired perception.

Index Terms—Web Accessibility; Autonomous Artificial Intelligence Agents; Automated Testing

I. INTRODUCTION

Web accessibility is central to inclusive software design, yet existing evaluation methods predominantly focus on code conformance, as illustrated by Web Content Accessibility Guidelines (WCAG) rule checks, often overlooking how individuals with disabilities actually navigate and perceive interfaces in practice [1]. Although these tools are useful for guidance and a first approach, they can be unable to thoroughly assess how these impairments affect real-time usability, task completion, or perception of interface elements. While often ignoring behavioral context; they do not simulate navigation, focus order, or sequential interaction flows that can significantly impact accessibility. Individuals with disabilities have diverse needs and experiences, many of which may not be fully addressed by level-A WCAG guidelines alone.

Although the access to information is a human right, the urgency of dynamic and human-centered accessibility evaluation is also supported by data. Approximately 2.2 billion people globally have some sort of visual impairment, including both near and distant vision issues [2]. In Colombia, among the estimated 2.65 million people living with a disability by the DANE, approximately 57% report that vision-related activities present the greatest challenges [3]. In bigger countries like the USA, the CDC reports that, about 5.5% of adults (nearly 19 million people) have blindness or serious difficulty seeing [4]. Furthermore, screen reader users find that the most problematic items to interact with in a webpage are CAPTCHA,

interactive elements, ambiguous links or buttons, unexpected screen changes, lack of keyboard support, among others [5]. Even more tellingly, these users extract information from data visualizations 61% less accurately and take 211% more time compared to sighted users [6].

We explore the possibility of approaching accessibility testing using autonomous AI agents capable of interacting with websites while being exposed to simulated visual impairments. These agents are prompted with specific user tasks (e.g., locating a button, submitting a form) and attempt to complete them. They can also integrate outputs from assistive technologies, for example, capturing a screen reader's textual narration.

This multi-modal input allows the agent to simulate how a user with various levels of visual impairment interacts with content, opening the door for dynamic automated testing other existing tools might be unable to uncover. For instance, detecting unuseful alt text, mislabeled controls or mismatches between rendered content and screen reader output.

This paper outlines our motivation, proposed approach, poses future evaluation research questions, and discusses implementation considerations for such a system.

II. MOTIVATION

The widespread adoption of the web has emphasized the importance of ensuring that online content is accessible to everyone, including individuals with various disabilities [7]. Despite this, less than 4% of the top one million websites are fully accessible. Over 96% contain detectable WCAG failures, with an average of 51 errors per page [8]. Common issues include missing alt text (present on 56% of home pages), low contrast, and poor form labeling [9].

Rigorous testing is integral to ensuring reliability of web applications. Usability testing that involves having real users interacting with a live web application, allows for the identification of issues in action, enables for iterative feedback and a more realistic assessment of how they might experience the interface. Effective testing improves satisfaction, avoids rework, and ensures genuine inclusive design [10].

Nevertheless, practitioners often find manual testing on people with impairments especially challenging, due to the difficulty of recruiting participants, communication, travelling to experiment location, among others [11].

Recent research highlights that most applications of AI in web accessibility focus on generating alternative text for images, automating compliance checks, suggesting corrections,

and creating alternative interfaces to improve access [12]. However, these approaches remain largely centered on white-box analysis and lack addressing the challenges of evaluating accessibility from the perspective of real user interaction.

While recent advances have demonstrated the potential of autonomous agents for usability testing [13], mobile accessibility feature testing [14] and accessibility testing [15], to our knowledge, these agents have not yet been applied to web accessibility evaluation in a multi-modal decision-based way. Bridging this gap offers the opportunity to deliver scalable, repeatable, and realistic assessments of how users with disabilities interact with web interfaces, potentially enhancing the detection and remediation of accessibility issues and making the evaluation process more efficient and comprehensive.

The objectives of this work are twofold. First, we aim to explore the use of perceptual filters and multimodal inputs, specifically applying blur to simulate different impairments; while also using screen reader outputs, to approximate the experience of real world users.

Second, we will analyze whether certain UI elements become inaccessible or difficult to perceive under these simulated conditions, and investigate if common layout structures or design patterns can fail in these situations. Through this approach, we seek to identify not only the direct effects of visual filters on usability, but also the broader structural and behavioral implications for accessible web design.

III. RELATED WORK

A. Web accessibility compliance

Conventional web accessibility assessments primarily rely on static checkers that verify compliance with WCAG criteria by inspecting HTML and CSS structures. Tools including WAVE by WebAIM [16] and IBM's NPM accessibility-checker [17] exemplify this approach. However, several studies have revealed significant limitations in these tools, noting they often lack semantic awareness and fail to consider user perspectives, leading to incomplete or misleading results [1].

For instance, Todorov et al. evaluated Bulgarian museum websites uncovered widespread accessibility failures despite technical correctness [18]. Similarly, Inal et al. conducted a study of Norwegian municipal websites and found that although legislation mandated WCAG compliance, issues such as low-quality alternative text persisted. Chiou et al. further observed that many common web accessibility issues arise in responsive sites during resizing, highlighting the importance of evaluating accessibility across different device viewports [19].

Complimentarily, WCAG conformance varies by level, from A (lower) to AAA (higher), therefore, a website labeled as compliant may still fall short of more specific accessibility standards. This underscores the need for real-time evaluations that reflect how visually impaired users interact with web interfaces. While automated tools are useful for early feedback, they are unable to identify all critical accessibility, functionality, and usability issues affecting this population [18].

B. Automated accessibility Testing

Recent studies introduce hybrid frameworks that combine guidelines with heuristic, automated, or AI-driven methods to improve accessibility evaluation. Watanabe et al. [20] show that machine learning can detect ARIA landmarks in web apps, revealing how classifiers may infer structure when key accessibility tags are missing. Similarly, models have been used to identify and remediate accessibility issues, helping sites align with accessibility standards. Some approaches analyze source code to suggest fixes [21], [22], while others operate on rendered pages using prompt-based methods [23]. Mehralian et al. [24] propose a rule-based system for testing dynamic content in Android apps, and Tafreshipour et al. [25] show that mutation testing can uncover additional errors by exploring app states and applying accessibility rules.

Dynamic testing, as described by Linares-Vásquez et al. [26], enables developers and stakeholders to verify and validate that running software behaves as expected. In this context, automated testing, especially when enhanced by machine learning and AI [13], [27], [28], is becoming increasingly popular and sophisticated, offering faster, more reliable, and more comprehensive accessibility evaluation.

C. Autonomous AI agents

The use of autonomous AI agents to simulate user interaction with web interfaces has also been explored. Lu et al. [13] introduce UXAgent; a notable system that uses LLM agents to mimic thousands of diverse user personas in web usability studies. The agents interact with live websites via browser automation, providing qualitative and quantitative feedback that supports iterative UX design.

Complementing this, a GitHub Copilot extension that proactively embeds accessibility guidance into the coding workflow was unveiled by Mowar et al. [29]. Their study shows how AI assistants can suggest accessible UI code, highlight missing attributes, and prompt manual verification during development.

While not strictly agent-based, Zhong et al. [15] leveraged large language models to identify discrepancies between screen reader outputs, providing a novel approach to detecting accessibility errors.

Another recent advancement in this area is AXNav by Taeb et al. [14], a system that interprets mobile accessibility test instructions written in natural language and executes them on remote cloud devices using an LLM-based multiagent planner. This approach demonstrates how LLM-driven agents can automate complex evaluations and provide actionable, context-rich feedback for developers.

Multimodal agents that adaptively present content based on user needs, transforming visual content into speech or simplified visuals for users with auditory or visual processing disorders were also proposed by Rajagopal et al. [30]. While not focused on web testing, their work shows how agents can model disability-specific interactions across modalities.

Collectively, these works [13], [27], [28] illustrate a shift toward agent-based, multimodal, and LLM-driven approaches
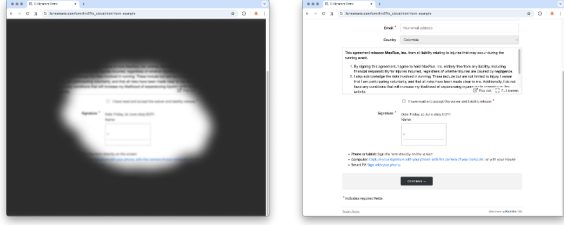
Fig. 1: Left: Glaucoma filter applied to an example form requiring a signature. The "Submit" button is no longer visible, making it difficult to locate. Right: Original version.
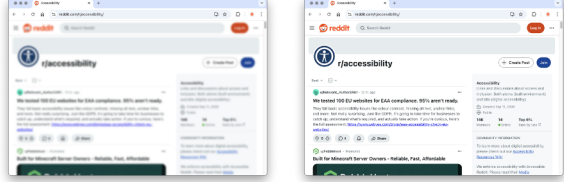

Fig. 2: Left: Myopia filter (-3 diopters) applied to a social media webpage, reducing clarity and edge sharpness. Right: Original version.

that move beyond static code analysis to more closely approximate real user experiences. Such agents can interact with web content without access to the underlying code, adapt based on feedback, and process rich multimodal inputs, enabling them to reason about both the visual layout and the spoken feedback of the user interface. While LLM agents excel at open-ended reasoning and can provide qualitative insights alongside quantitative logs, they require careful prompting and can be slower or less predictable.

IV. SYSTEM DESIGN

A. Visual Filtering

The simulations shown on Figs. 1 and 2 were implemented using the Visual Impairments Simulator Chrome extension [31]. Simulating visual impairments through perceptual filters allows us to have an approximation to the experiences of users with vision loss conditions. Still, these tools are inherently limited; since visual disabilities are highly individualized, and no filter can perfectly replicate the subjective visual experience of every user.

In light of these considerations, the beta testing of this agent will focus on a defined set of visual impairments. The prototype includes simulations of glaucoma (peripheral field loss and tunnel vision [32]), diabetic retinopathy (scattered blind spots and patchy visual fields [32]), cataracts (global blurriness and reduced contrast sensitivity [32]), and myopia (blur at distance with varying levels of diopter correction [32]). These conditions were selected as representative examples because they encompass distinct perceptual characteristics, in ways that reflect known physiological effects.

In addition, experimenting with different viewports, font sizes, and other browser configurations [19] will be done to test how these factors affect usability.
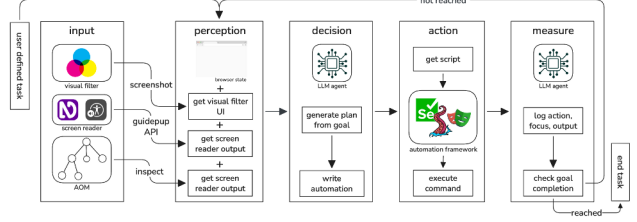

Fig. 3: Proposed pipeline for the agent feeding inputs

Future work involves collaboration with ophthalmologists and vision science experts to improve the clinical accuracy of these filters. Their insight can guide the calibration of filter parameters and help us design simulations that more closely resemble the lived experiences of users.

B. Assistive output integration

The agent must capture what an assistive technology user would hear and do. For screen reader output, we can integrate a driver API, such as Guidepup [33] that programmatically controls VoiceOver [34] or NVDA [35]. This driver issues the same keyboard commands a user would and exposes the spoken utterances [33]. The agent framework would invoke this API at each step and record the resulting string of text (including element role, name, state) as sensory input.

For navigation logs, the agent can log focus events. As the agent presses Tab, arrow keys, click, enter, etc., we can attach listeners to record each focus transition and action. This yields a sequence of elements visited in order, that can then be executed using a browser automation framework, namely Playwright [36], Selenium [37], or Kraken [38].

The agent can also log accessibility metadata, thereby having the ability to cross-check perception. This means reading the ARIA attributes for the current focused element. For instance, when an element is in focus, the agent can query its ARIA label, via the Accessibility Object Model (AOM) or other APIs. This is then compared to the other information (visual, audio). If the visible text of a button says "Search" but its ARIA label is "Submit" that mismatch is noteworthy. Altogether, ARIA attributes and accessibility metadata enrich the logs by providing a semantic model of user actions and interface states, allowing for more precise analysis and how accessibility features are utilized or bypassed. This is useful for verifying consistency, which is one of the guidelines on WCAG and IBM's [39].

In general, the resulting logs constitute a detailed record of the agent's navigation path, enabling systematic analysis of interaction patterns. They can be examined to identify instances where interface elements were bypassed, or to detect deviations from the expected focus order. Furthermore, replaying recorded sessions facilitates a nuanced assessment of element discoverability, the interpretation of screen reader output, and the factors influencing task completion or failure. Together, these multimodal data streams comprehensively characterize the agent's perceptual state at each interaction step.

C. Decision and Action Module

Our approach consists of prompting the agent with a basic user goal. E.g. "given this screenshot and screen reader transcript, where is the `Submit` button?". The agent will then attempt to complete the task, using the screen reader and seeing through the visual impairment filter applied to the interface. Consider one scenario which may involve an AI agent attempting to find and click a `Submit` button after filling out a form. With a glaucoma filter applied (see Fig. 1), the button may become difficult to distinguish due to peripheral blur or low contrast. The agent has to decide which interaction method works best (pointer, keyboard, etc) and execute the decision.

D. Workflow Definition

Each testing task needs to be defined in advance. This can be done either with a script or using natural language to define a goal. Then, the agent will execute said task in a closed loop that has a completion or failure condition. All in all, the loop can be structured in this way (see Fig. 3).

Steps: Perception (capture current state: screenshot, screen-reader output, optionally AOM information) → Decision (agent chooses next action, e.g., LLM generates plan and writes a rule-based script that framework follows) → Action (execute step) → Measure (log outputs, check goal, repeat).

E. Metrics and Analysis

After execution is complete, some of the metrics we propose that the system displays are both classic usability and accessibility-specific. These include the agent's task success rate, efficiency (measured by completion time or number of actions), and the frequency of interaction errors, missed clicks or incorrect actions that require backtracking.

We also include the consistency between screen reader output and the visible UI, flagging any mismatches between spoken labels and on-screen text or roles. Visual robustness can also be examined by analyzing the webpage before the filter, discovering layout faults like overlapping or off-screen elements, among others.

Heuristic checks are performed during testing, among them are verifying that text scales appropriately in high-contrast or large-text modes and monitoring for navigation loops. Throughout the process, we aggregate logs of the agent's actions and screen reader output for offline analysis, enabling manual review or further explanation by the agent. By comparing these metrics across different simulated conditions, we can quantify the impact of visual impairments on usability and identify both layout and semantic accessibility issues.

V. Evaluation

To investigate the feasibility and effectiveness of using autonomous AI agents for accessibility evaluation, we pose several research questions. We ask whether autonomous AI agents can realistically emulate the interaction patterns of users with vision-related impairments when navigating web interfaces. This includes examining which visual filters or perceptual constraints most effectively represent different types of visual impairments in a simulated context, and what design principles are necessary for agents to approximate visual impairments through perception-based (rather than code-aware) interaction.

We also explore whether these agents can surface accessibility problems that static tools overlook, like poor contrast visibility, confusing focus order, or dynamic content that is not screen-reader friendly, to name a few. Another important question is whether agent-based testing, when combined with simulated impairments, can produce accessibility assessments that are reliable and generalizable. We are interested in the extent to which these systems can internalize accessibility heuristics through observation of human interaction data, rather than relying solely on static rule sets.

Finally, we investigate how combining simulated visual impairment filters with screen reader output affects the agent's ability to detect accessibility issues, and whether multi-modal input can uncover problems, say, missing alt text, that would not surface if using only visual filtering.

To evaluate our approach, we will conduct experiments on a curated set of benchmark web pages, including public sites with documented accessibility issues. Inspired by Alameer et al. [40], who compiled websites with internationalization challenges, we may extend our benchmarks to include similar cases. For each page, we will define representative tasks, and assess agent performance across different visual filters and input modalities. We plan to complement these quantitative results with qualitative feedback from accessibility experts and targeted user studies, to further validate the effectiveness and generalizability of agent-based accessibility testing.

VI. Conclusion & Future Work

This paper proposes a new direction for accessibility evaluation: the use of autonomous, multi-modal AI agents that simulate visual impairments and interact with web content through perception, not code. By combining visual filters, screen reader output, and task-based prompting, this approach aims to approximate the experiences of users with diverse visual disabilities and to surface accessibility issues that may be missed by static tools.

The potential of this method lies in its ability to reveal failures that only emerge under perceptual constraints, and to provide richer, multi-modal insights into web accessibility. At the same time, we recognize the methodological challenges of using simulated agents as proxies for real users, and stress the importance of human validation.

Future work will involve comparing this approach with existing methods, such as seeking consultation services from accessibility experts, comparison with real user evaluations and static methods. Together with a detailed analysis of system performance and cost-effectiveness, weighing the benefits of automation against the increased computational demands. Ultimately, we hope this research will lay the groundwork for more dynamic, user-centered accessibility evaluation methods and inspire further research in this area.

REFERENCES

[1] J. Ara, C. Sik-Lanyi, A. Kelemen, and T. Guzsvinecz, "An inclusive framework for automated web content accessibility evaluation," *Universal Access in the Information Society*, pp. 1–27, 2024.

[2] World Health Organization (WHO), "World report on vision," https://www.who.int, 2023.

[3] Departamento Administrativo Nacional de Estadística (DANE), "Estado actual de la medición de la discapacidad en colombia," 2022, pDF Presentation. [Online]. Available: https://www.dane.gov.co/files/investigaciones/notas-estadisticas/abr_2022_nota_estadistica_Estado%20actual_de_la_medicion_de_discapacidad_en%20Colombia_presentacion.pdf

[4] US Centers for Disease Control and Prevention (CDC), "Disability impacts all of us infographic," https://www.cdc.gov/disability-and-health/articles-documents/disability-impacts-all-of-us-infographic.html, 2025.

[5] WebAIM. (2025) Webaim: Screen reader survey results. Accessed: 2025-06-24. [Online]. Available: https://webaim.org/projects/screenreadersurvey10/

[6] J. O. Wobbrock *et al.*, "Understanding screen-reader users' experiences with online data visualizations," in *ASSETS '21: The 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, 2021, p. 16.

[7] I. Abu Doush, K. Sultan, M. A. Al-Betar, Z. Almeraj, Z. A. A. Alyasseri, and M. A. Awadallah, "Web accessibility automatic evaluation tools: to what extent can they be automated?" *CCF Transactions on Pervasive Computing and Interaction*, vol. 5, no. 3, pp. 288–320, 2023.

[8] WebAIM. (2025) Webaim: The webaim million. Accessed: 2025-06-24. [Online]. Available: https://webaim.org/projects/million/

[9] AudioEye, "Web accessibility stats and data 2024," https://www.audioeye.com/post/accessibility-statistics/, 2024.

[10] AccessDesign Studio, "Beyond resolutions: Why 2025 is the year to invest in website accessibility," https://accessdesignstudio.com/beyond-resolutions-why-2025-is-the-year-to-invest-in-website-accessibility/, 2025.

[11] W. Xu, C. Anderton, K. Weir, and A. Theil, "Conducting vr user studies with people with vision/hearing impairments: Challenges and mitigation strategies," *arXiv preprint arXiv:2504.07256*, 2025.

[12] G. Vera-Amaro and J. R. Rojano-Cáceres, "Towards accessible website design through artificial intelligence: A systematic literature review," *Information and Software Technology*, p. 107821, 2025.

[13] Y. Lu, B. Yao, H. Gu, J. Huang, J. Wang, Y. Li, J. Gesi, Q. He, T. J.-J. Li, and D. Wang, "Uxagent: A system for simulating usability testing of web design with llm agents," *arXiv preprint arXiv:2504.09407*, 2025.

[14] M. Taeb, A. Swearngin, E. Schoop, R. Cheng, Y. Jiang, and J. Nichols, "Axnav: Replaying accessibility tests from natural language," in *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 2024, pp. 1–16.

[15] M. Zhong, R. Chen, X. Chen, J. Fogarty, and J. O. Wobbrock, "Screenaudit: Detecting screen reader accessibility errors in mobile apps using large language models," in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 2025, pp. 1–19.

[16] WebAIM, "Wave web accessibility evaluation tools," https://wave.webaim.org/, 2025, suite of tools (browser extension, API, AIM report) for WCAG/Section 508 evaluation.

[17] IBM, "accessibility-checker," NPM module, 2025, node.js automated accessibility testing tool (Puppeteer, Selenium, etc.).

[18] T. Todorov, G. Bogdanova, and M. Todorova-Ekmekci, "Accessibility of bulgarian regional museums websites," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 3, 2022.

[19] P. T. Chiou, R. Winn, A. S. Alotaibi, and W. G. Halfond, "Automatically detecting reflow accessibility issues in responsive web pages," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, 2024, pp. 1–13.

[20] W. M. Watanabe, G. de Lemos, and R. W. Nascimento, "Accessibility landmarks identification in web applications based on dom elements classification," *Universal Access in the Information Society*, vol. 23, no. 2, pp. 765–777, 2024.

[21] V. Ramineni, B. S. Ingole, A. R. Banarse, M. S. Krishnappa, N. K. Pulipeta, and V. Jayaram, "Leveraging ai and machine learning to address ada non-compliance in web applications: A novel approach to enhancing accessibility," in *2024 Third International Conference on Artificial Intelligence, Computational Electronics and Communication System (AICECS)*. IEEE, 2024, pp. 1–6.

[22] K. Kuszczyński and M. Walkowski, "Comparative analysis of open-source tools for conducting static code analysis," *Sensors*, vol. 23, no. 18, p. 7978, 2023.

[23] Z. He, S. F. Huq, and S. Malek, "Enhancing web accessibility: Automated detection of issues with generative ai," *Proceedings of the ACM on Software Engineering*, vol. 2, no. FSE, pp. 2264–2287, 2025.

[24] F. Mehralian, Z. He, and S. Malek, "Automated accessibility analysis of dynamic content changes on mobile apps," in *IEEE/ACM 47th International Conference on Software Engineering (ICSE)*, 2025, pp. 1–13.

[25] M. Tafreshipour, A. Deshpande, F. Mehralian, I. Ahmed, and S. Malek, "Ma11y: A mutation framework for web accessibility testing," in *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2024, pp. 100–111.

[26] M. Linares-Vásquez, K. Moran, and D. Poshyvanyk, "Continuous, evolutionary and large-scale: A new perspective for automated mobile app testing," *arXiv preprint arXiv:1801.06267*, 2018.

[27] M. Lanham, "Getting started with ai agents," in *AI Agents in Action*. Shelter Island, NY: Manning Publications, 2025, ch. 1.

[28] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin *et al.*, "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, 2024.

[29] P. Mowar, Y.-H. Peng, J. Wu, A. Steinfeld, and J. P. Bigham, "Codea11y: Making ai coding assistants useful for accessible web development," in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 2025, pp. 1–15.

[30] A. Rajagopal, V. Nirmala, I. J. Jebadurai, A. M. Vedamanickam, and P. U. Kumar, "Design of generative multimodal ai agents to enable persons with learning disability." in *ICMI Companion*, 2023, pp. 259–271.

[31] airadavometra, "Simulator of visual impairments," https://chromewebstore.google.com/detail/simulator-of-visual-impai/imohhjdbajiihdogpnphgbocfodbofip, 2023, chrome Extension, Version 1.0.1, accessed: 2025-06-26.

[32] G. H. Cassel, *The Eye Book: A Complete Guide to Eye Disorders and Health*, 2nd ed. Baltimore, MD: Johns Hopkins University Press, 2021.

[33] Guidepup Team, "Guidepup: Screen reader driver for test automation," https://www.guidepup.dev, 2025.

[34] Apple Inc., "Voiceover for macos," 2024. [Online]. Available: https://support.apple.com/guide/voiceover/welcome/mac

[35] N. Access, "Nvda: Nonvisual desktop access," 2024. [Online]. Available: https://www.nvaccess.org

[36] Microsoft Corporation, "Playwright: Open-source browser automation," urlhttps://playwright.dev/, 2025.

[37] B. García, C. D. Kloos, C. Alario-Hoyos, and M. Munoz-Organero, "Selenium-jupiter: A junit 5 extension for selenium webdriver," *arXiv preprint arXiv:2402.01480*, 2024.

[38] W. Ravelo-Méndez, C. Escobar-Velásquez, and M. Linares-Vásquez, "Kraken 2.0: A platform-agnostic and cross-device interaction testing tool," *Science of Computer Programming*, vol. 225, p. 102897, 2023.

[39] IBM Corporation, "Ibm accessibility: Equal access toolkit," https://www.ibm.com/able/toolkit/verify, 2025.

[40] A. Alameer, S. Mahajan, and W. G. Halfond, "Detecting and localizing internationalization presentation failures in web applications," in *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2016, pp. 202–212.